ชั้นวางเรื่องราว

โดย

นาย สิรวิชญ์ ทิมสุวรรณ

นางสาว ณิชาภัทร ชมภูน้อย

โครงงานพิเศษนี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตร

วิทยาศาสตรบัณฑิต

สาขาวิชาวิทยาการคอมพิวเตอร์

คณะวิทยาศาสตร์และเทคโนโลยี มหาวิทยาลัยธรรมศาสตร์

ปีการศึกษา 2567

ชั้นวางเรื่องราว

โดย

นาย สิรวิชญ์ ทิมสุวรรณ
นางสาว ณิชาภัทร ชมภูน้อย

โครงงานพิเศษนี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตร
วิทยาศาสตรบัณฑิต
สาขาวิชาวิทยาการคอมพิวเตอร์
คณะวิทยาศาสตร์และเทคโนโลยี มหาวิทยาลัยธรรมศาสตร์
ปีการศึกษา 2567

Taledge

BY

Mr.SIRAWICH TIMSUWAN
Ms.NICHAPAT CHOMPOONOI

A FINAL-YEAR PROJECT REPORT SUBMITTED IN PARTIAL
FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF
BACHELOR OF SCIENCE
COMPUTER SCIENCE
FACULTY OF SCIENCE AND TECHNOLOGY
THAMMASAT UNIVERSITY
ACADEMIC YEAR 2024

มหาวิทยาลัยธรรมศาสตร์
คณะวิทยาศาสตร์และเทคโนโลยี

รายงานโครงงานพิเศษ

ของ

นาย สิรวิชญ์ ทิมสุวรรณ
นางสาว ณิชาภัทร ชมภูน้อย

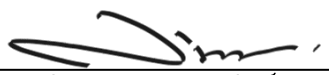
เรื่อง

ชั้นวางเรื่องราว

ได้รับการตรวจสอบและอนุมัติ ให้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตร
หลักสูตรวิทยาศาสตรบัณฑิต สาขาวิชาวิทยาการคอมพิวเตอร์
เมื่อ วันที่ 30 พฤษภาคม พ.ศ. 2568

อาจารย์ที่ปรึกษา 
(อาจารย์ สิริกันยา นิลพานิช)

กรรมการสอบโครงงานพิเศษ 
(ผศ. ดร.เสาวลักษณ์ วรรธนาภา)

กรรมการสอบโครงงานพิเศษ 
(ผศ. ดร.ลัมพาพรรณ พันธุ์ชูจิตร์)

มหาวิทยาลัยธรรมศาสตร์
คณะวิทยาศาสตร์และเทคโนโลยี

รายงานโครงงานพิเศษ

ของ

นาย สิรวิชญ์ ทิมสุวรรณ
นางสาว ณิชาภัทร ชมภูน้อย

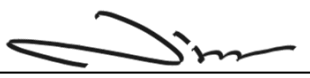
เรื่อง

ชั้นวางเรื่องราว

ได้รับการตรวจสอบและอนุมัติ ให้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตร
หลักสูตรวิทยาศาสตรบัณฑิต สาขาวิชาวิทยาการคอมพิวเตอร์
เมื่อ วันที่ 30 พฤษภาคม พ.ศ. 2568

อาจารย์ที่ปรึกษา 
(อาจารย์ สิริกันยา นิลพานิช)

กรรมการสอบโครงงานพิเศษ 
(ผศ. ดร.เสาวลักษณ์ วรรธนาภา)

กรรมการสอบโครงงานพิเศษ 
(ผศ. ดร.ลัมพาพรรณ พันธุ์ชูจิตร์)

หัวข้อโครงงานพิเศษ	ชั้นวางเรื่องราว
ชื่อผู้เขียน	นาย สิรวิชญ์ ทิมสุวรรณ
ชื่อผู้เขียน	นางสาว ณิชาภัทร ชมภูน้อย
ชื่อปริญญา	วิทยาศาสตรบัณฑิต สาขาวิชาวิทยาการคอมพิวเตอร์
สาขาวิชา/คณะ/มหาวิทยาลัย	สาขาวิชาวิทยาการคอมพิวเตอร์ คณะวิทยาศาสตร์และเทคโนโลยี มหาวิทยาลัยธรรมศาสตร์
อาจารย์ที่ปรึกษาโครงงานพิเศษ	อาจารย์ สิริกันยา นิลพานิช
ปีการศึกษา	2567

บทคัดย่อ

การเขียนนิยายที่มีพล็อตเรื่องที่น่าสนใจและชัดเจนเป็นสิ่งสำคัญที่ช่วยให้นักเขียนสามารถสร้างผลงานที่มีคุณภาพ อย่างไรก็ตาม การวางพล็อตที่ซับซ้อนและต้องการการจัดระเบียบข้อมูลอย่างเป็นระบบยังคงเป็นความท้าทายสำหรับนักเขียนหลายคน

โครงงาน “TALEDGE: ชั้นวางเรื่องราว” เว็บแอพพลิเคชั่นสำหรับช่วยในการวางพล็อตนิยาย จึงได้ถูกพัฒนาขึ้นเพื่อช่วยนักเขียนในกระบวนการวางพล็อตเรื่อง โดยระบบมีฟังก์ชันที่ช่วยในการสร้างโครงสร้างพล็อต การจัดการตัวละคร และการวางเหตุการณ์ในเรื่อง

ด้วยการออกแบบและพัฒนาความสามารถของเว็บแอพพลิเคชันทั้งหมดนี้ ผู้พัฒนาหวังเป็นอย่างยิ่งว่า เว็บแอพพลิเคชันช่วยในการวางพล็อตนิยายนี้ จะสามารถช่วยนักเขียนในการจัดระเบียบและออกแบบพล็อตเรื่องได้อย่างเป็นระบบและสามารถใช้งานได้จริงตรงตามจุดประสงค์ที่วางไว้

คำสำคัญ: วางพล็อตนิยาย, นักเขียน, เว็บแอพพลิเคชั่น, การเขียนนิยาย

Thesis Title	Taledge
Author	Mr. Sirawich Timsuwan
Author	Ms. Nichapat Chompoonoi
Degree	Bachelor of Science
Major Field/Faculty/University	Computer Science Faculty of Science and Technology Thammasat University
Project Advisor	Sirikunya Nilpanich
Academic Years	2024

ABSTRACT

Writing a novel with a clear and engaging plot is essential for authors to create high-quality works. However, crafting a complex plot and organizing its elements systematically remain challenging tasks for many writers.

The project “TALEDGE: The Story Organizer”, a web application for novel plot design, has been developed to assist writers in structuring their plots effectively. The system features tools for creating plot structures, managing characters, and organizing story events.

With the design and functionality of this web application, the developers hope that it will support writers in systematically organizing and designing their story plots. Additionally, the application aims to meet its intended purpose and be a practical tool for real-world use.

Keywords: plot design, writers, web application, novel writing

กิตติกรรมประกาศ

โครงงานเรื่อง “TALEDGE: ชั้นวางเรื่องราว” เว็บแอพพลิเคชั่นสำหรับช่วยในการวางพล็อตนิยายิ จัดทำขึ้นเพื่อการศึกษาของนักศึกษาระดับปริญญาตรี โครงงานสามารถดำเนินการสำเร็จได้ เนื่องจากได้รับความอนุเคราะห์และการสนับสนุนเป็นอย่างดีจากอาจารย์ที่ปรึกษาโครงงาน อาจารย์ สิริกันยา นิลพานิช คณาจารย์คณะวิทยาศาสตร์และเทคโนโลยี สาขา วิทยาการคอมพิวเตอร์ ที่ได้กรุณาให้คำปรึกษา ความรู้ ข้อคิด และคำแนะนำ เพื่อปรับปรุง แก้ไขข้อบกพร่อง จนกระทั่งโครงงานนี้สำเร็จลุล่วงไปได้ด้วยดี ผู้จัดทำโครงงานขอกราบขอบพระคุณเป็นอย่างสูงไว้ ณ ที่นี้

สุดท้ายนี้ ผู้จัดทำโครงงานหวังเป็นอย่างยิ่งว่าโครงงานนี้จะเป็นประโยชน์สำหรับนักเขียนและผู้ที่สนใจในวงการวรรณกรรมต่อไป

คณะผู้จัดทำ

นาย สิรวิชญ์ ทิมสุวรรณ

นางสาว ณิชาภัทร ชมภูน้อย

สารบัญ

หน้า

บทคัดย่อ 2

ABSTRACT 4

กิตติกรรมประกาศ 5

สารบัญ **Error! Bookmark not defined.**

สารบัญตาราง **Error! Bookmark not defined.**

สารบัญภาพ **Error! Bookmark not defined.**

รายการสัญลักษณ์และคำย่อ **Error! Bookmark not defined.**

บทที่ 1 บทนำ 1

1.1 ความเป็นมาและความสำคัญของโครงงาน 1

1.2 วัตถุประสงค์ 2

1.3 ขอบเขตของโครงงาน 2

1.4 ประโยชน์ของโครงงาน 2

1.5 ข้อจำกัดของโครงงาน 2

บทที่ 2 วรรณกรรมและงานวิจัยที่เกี่ยวข้อง 3

2.1 แนวคิดทฤษฎีที่เกี่ยวข้อง 3

2.1.1 Content Organization and Narrative Theory 3

2.1.2 แนวคิดการออกแบบ UX/UI (User Experience and Interface Design) 4

2.1.3 ทฤษฎีฐานข้อมูลและการจัดการข้อมูล (Database and Knowledge Management) 5

2.2 เทคโนโลยีที่เกี่ยวข้อง 6

2.2.1 การพัฒนา Web Application ในส่วนของการ Coding 6

2.2.2 Cloud Services 9

2.2.3 CMS (Content Management System) 11

2.2.4 API 11

2.2.5 ระบบพจนานุกรม 13

บทที่ 3 วิธีการวิจัย 16

3.1 ภาพรวมของโครงงาน 16

3.2 การวิเคราะห์ขอบเขตและความต้องการของระบบ 17

3.3 การออกแบบระบบ 34

3.4 ประเด็นที่น่าสนใจและท้าทาย 42

3.5 ผลลัพธ์ที่คาดหวัง 43

บทที่ 4 ผลการดำเนินงาน 44

4.1 หน้าหลัก (Home) 44

4.2 โปรเจกต์ที่เคยสร้าง 45

4.3 หน้าสร้างพล็อตและจัดการเหตุการณ์ (Timeline) 45

4.4 การจัดการตัวละคร (Characters Panel) 46

4.5 การจัดกลุ่มตัวละคร (Character Groups) 46

4.6 การจัดการไอเทม (Items Panel) 47

4.7 การเขียนต้นฉบับ (Manuscript Panel) 47

4.8 การจัดการบท (Chapter Directory) 48

4.9 การเชื่อมโยงความสัมพันธ์รหว่างตัวละคร 48

4.10 การจัดการโลกในเรื่อง (World Panel) 49

4.11 การจัดการไอเดียหรือการอ้างอิง (Research Panel) 49

4.12 ระบบ System (ผังระบบความสัมพันธ์หรือชนชั้น) 50

บทที่ 5 สรุป 51

5.1 สรุปผลการดำเนินงาน 51

5.2 ข้อเสนอแนะและแนวทางการพัฒนาต่อไป 52

รายการอ้างอิง 53

ภาคผนวก 55

ภาคผนวก ก. ส่วนของ Relationships หรือ แผนผังความสัมพันธ์ 55

ภาคผนวก ข. ส่วนของ System หรือ ระบบชนชั้น 56

ภาคผนวก ค. การใช้ API Dictionary app.py 58

บทที่ 1
บทนำ

1.1 ความเป็นมาและความสำคัญของโครงงาน

ในยุคปัจจุบัน การเขียนนิยายได้รับความนิยมอย่างแพร่หลาย ทั้งในรูปแบบสิ่งพิมพ์และแพลตฟอร์มออนไลน์ เช่น เว็บไซต์และแอปพลิเคชันสำหรับนักเขียน อย่างไรก็ตาม การสร้างนิยายที่มีโครงเรื่องที่น่าสนใจและสอดคล้องมักเป็นปัญหาสำคัญของนักเขียน โดยเฉพาะผู้เริ่มต้นหรือผู้ที่ต้องการพัฒนาแนวคิดให้ชัดเจนและมีประสิทธิภาพมากขึ้น

หนึ่งในความท้าทายหลักของนักเขียน คือการจัดระเบียบข้อมูลเกี่ยวกับพล็อต ตัวละคร สถานที่ และเหตุการณ์ต่าง ๆ ในเรื่อง การวางพล็อตที่ไม่มีระบบอาจทำให้เกิดความสับสนและเสียเวลาในการปรับเปลี่ยนเนื้อหา

ด้วยเหตุนี้ โครงงาน “TALEDGE: ชั้นวางเรื่องราว” เว็บแอพพลิเคชั่นสำหรับช่วยในการวางพล็อตนิยาย จึงถูกพัฒนาขึ้นเพื่อเป็นเครื่องมือที่ช่วยนักเขียนในการจัดระเบียบและออกแบบพล็อตเรื่องได้อย่างเป็นระบบ โดยเน้นการใช้งานง่ายและตอบโจทย์ความต้องการของนักเขียนทุกระดับ

แอพพลิเคชั่นนี้ไม่เพียงช่วยลดความซับซ้อนในการวางพล็อต แต่ยังช่วยเพิ่มประสิทธิภาพและความคิดสร้างสรรค์ให้นักเขียนสามารถพัฒนาเนื้อหาได้อย่างราบรื่น ซึ่งถือเป็นการสนับสนุนวงการวรรณกรรมในยุคดิจิทัลให้เติบโตอย่างยั่งยืน

1.2 วัตถุประสงค์

โครงงานนี้มีเป้าหมายเพื่อสร้างเว็บแอปพลิเคชันสำหรับการวางพล็อตนิยาย และผู้จัดทำได้มีการกำหนดวัตถุประสงค์เพื่อให้บรรลุตามเป้าหมายไว้ ดังนี้

1. เพื่อพัฒนาเว็บแอพพลิเคชั่นสำหรับช่วยในการวางพล็อตนิยาย
2. เพื่อช่วยนักเขียนจัดโครงเรื่อง ตัวละคร และเหตุการณ์ในนิยาย
3. เพื่อช่วยลดเวลาและความยุ่งยากในการวางพล็อต

1.3 ขอบเขตของโครงงาน

ระบบจะช่วยนักเขียนในการสร้างและจัดการพล็อตนิยาย โดยเน้นการใช้งานที่ง่าย รองรับการปรับแต่งและจัดเก็บข้อมูลต่าง ๆ เช่น ตัวละคร สถานที่ และเหตุการณ์

1.4 ประโยชน์ของโครงงาน

ในการดำเนินโครงงาน หากบรรลุตามเป้าหมายแล้ว จะเกิดประโยชน์จากการใช้งานเว็บแอปพลิเคชัน ดังนี้

1. สามารถจัดโครงสร้างเรื่องราว วางพล็อต และสร้างความต่อเนื่องของเนื้อหาได้ง่ายขึ้น
2. สามารถจัดเรียงความสัมพันธ์ระหว่างตัวละครได้อย่างเป็นระบบ
3. สามารถจัดเรียงไทม์ไลน์สำหรับติดตามเหตุการณ์ตามลำดับเวลา ทำให้วางแผนการเดินเรื่องและจุดพลิกผันของเรื่องได้ง่าย
4. สามารถจัดเก็บและปรับแต่งข้อมูลคาแรคเตอร์สำหรับตัวละครต่างๆ
5. สามารถจัดระเบียบข้อมูลนิยายได้ดียิ่งขึ้น

1.5 ข้อจำกัดของโครงงาน

1. เนื่องจากเป็นเว็บแอพพลิเคชันจึงไม่สามารถดาวน์โหลดลงเครื่องได้ หากต้องการเข้าใช้งานจะต้องเข้าใช้ผ่าน Browser ที่มีในตัวเครื่องเท่านั้น เช่น Google chrome, Safari หรืออื่น ๆ
2. เนื่องจากเป็นการใช้งานผ่านหน้า Browser ดังนั้น จะสามารถใช้งานได้ก็ต่อเมื่อมีการเชื่อมต่อระบบอินเทอร์เน็ตเท่านั้น

บทที่ 2
วรรณกรรมและงานวิจัยที่เกี่ยวข้อง

ในการจัดทำแอปพลิเคชันในการช่วยนักเขียน ผู้จัดทำได้ทำการศึกษาหลักการ แนวคิด ทฤษฎี เอกสารและงานวิจัยที่เกี่ยวข้องดังนี้

2.1 แนวคิดทฤษฎีที่เกี่ยวข้อง

2.1.1 Content Organization and Narrative Theory

การออกแบบโครงสร้างเนื้อหาและทฤษฎีการเล่าเรื่อง เป็นสองแนวคิดที่มีความสำคัญในการสร้างประสบการณ์ที่มีความหมายและน่าสนใจในการสื่อสาร โดยการออกแบบโครงสร้างเนื้อหาจะเน้นการจัดระเบียบข้อมูลให้เข้าใจง่ายและเข้าถึงได้สะดวก เช่น การแบ่งเนื้อหาตามลำดับความสำคัญ การใช้หัวข้อย่อย และการสร้างระบบนำทางที่ช่วยให้ผู้ใช้ค้นหาข้อมูลได้อย่างรวดเร็ว ส่วนทฤษฎีการเล่าเรื่องจะเน้นการสร้างเรื่องราวที่มีความน่าสนใจ โดยการจัดเรียงเหตุการณ์และการพัฒนาตัวละครที่เชื่อมโยงกับความตึงเครียดในเรื่องราว เพื่อดึงดูดความสนใจของผู้ฟังหรือผู้ชมให้ติดตามต่อไปจนจบ

ทั้งสองแนวคิดนี้มีความสัมพันธ์กันในการสร้างเนื้อหาที่ไม่เพียงแค่เข้าใจง่าย แต่ยังมีความน่าสนใจและสามารถเชื่อมโยงกับผู้ใช้ได้อย่างมีประสิทธิภาพ โดยการออกแบบโครงสร้างเนื้อหาจะช่วยให้การเล่าเรื่องเกิดขึ้นได้อย่างราบรื่นและไม่สับสน การจัดระเบียบเนื้อหาจึงเป็นการสร้างพื้นฐานให้กับการเล่าเรื่องที่มีคุณภาพ ขณะเดียวกันการใช้ทฤษฎีการเล่าเรื่องจะทำให้เนื้อหามีมิติ และช่วยเพิ่มความรู้สึกมีส่วนร่วมของผู้ชมที่สามารถสร้างความประทับใจและความสัมพันธ์กับเนื้อหานั้น

ในทางปฏิบัติ การนำทั้งสองแนวคิดมาผสมผสานจะทำให้การสร้างเนื้อหามีประสิทธิภาพมากขึ้น เช่น การออกแบบเว็บไซต์หรือแอปพลิเคชันที่มีการจัดระเบียบเนื้อหาชัดเจน พร้อมทั้งใช้การเล่าเรื่องในการเพิ่มความน่าสนใจให้กับผู้ใช้ หรือในกรณีของการเขียนบทภาพยนตร์ การจัดเรียงโครงสร้างเนื้อหาให้เหมาะสม

กับการพัฒนาเรื่องราวและตัวละคร จะช่วยให้เรื่องราวนั้นมีความต่อเนื่องและดึงดูดผู้ชมไปจนถึงจุดจบที่มีความหมาย

2.1.2 แนวคิดการออกแบบ UX/UI (User Experience and Interface Design)

UX (User Experience) ต้องเริ่มจากการเข้าใจความต้องการของผู้ใช้ที่แท้จริง โดยมุ่งเน้นไปที่การแก้ปัญหาและทำให้การใช้งานเป็นไปอย่างราบรื่น แนวคิดสำคัญคือการทำให้การเดินทางของผู้ใช้ตอบโจทย์และใช้งานง่าย เช่น การลดขั้นตอนที่ไม่จำเป็น การทำให้ผู้ใช้เข้าถึงข้อมูลหรือฟังก์ชันสำคัญได้รวดเร็ว และการออกแบบที่ช่วยให้ผู้ใช้รู้สึกมั่นใจในทุกการกระทำ การวิเคราะห์พฤติกรรมผู้ใช้และการทดสอบประสบการณ์ใช้งานจริงเป็นสิ่งที่ช่วยยืนยันว่าดีไซน์ของเราทำงานได้ดีในทุกสถานการณ์

UI (User Interface) มีบทบาทสำคัญในการทำให้ประสบการณ์ใช้งานราบรื่น โดย UI ไม่เพียงต้องดึงดูดสายตาแต่ยังต้องทำให้การโต้ตอบเป็นเรื่องง่าย แนวคิดคือการใช้สี รูปแบบตัวอักษร และองค์ประกอบกราฟิกที่สอดคล้องกัน สร้างความรู้สึกที่กลมกลืนและเป็นมิตรต่อผู้ใช้ การจัดลำดับความสำคัญของข้อมูลและการออกแบบพื้นที่ว่างยังช่วยให้ผู้ใช้โฟกัสกับสิ่งที่สำคัญที่สุดได้อย่างง่ายดาย

ซึ่ง UX/UI ที่ดีควรตอบสนองต่อความหลากหลายของอุปกรณ์และบริบทการใช้งาน เช่น ออกแบบให้เข้ากับหน้าจอขนาดต่างๆและปรับฟังก์ชันให้เหมาะสมกับสถานการณ์ เช่น การออกแบบแอปพลิเคชันที่ใช้งานง่ายทั้งในสภาพแสงจ้าและแสงน้อย หรือการเพิ่มฟีเจอร์เสียงเพื่อช่วยผู้ใช้ที่ไม่สะดวกมองหน้าจอ การวางแผนเชิงลึกและการปรับปรุงดีไซน์อย่างต่อเนื่องช่วยให้ผลิตภัณฑ์ตอบสนองความต้องการที่เปลี่ยนแปลงของผู้ใช้ได้อย่างมีประสิทธิภาพ

2.1.3 ทฤษฎีฐานข้อมูลและการจัดการข้อมูล (Database and Knowledge Management)

ข้อมูลเป็นทรัพยากรที่สำคัญที่สุดขององค์กรโดยทฤษฎีฐานข้อมูลและการจัดการข้อมูล มีบทบาทสำคัญในการช่วยให้องค์กรสามารถจัดเก็บและบริหารจัดการข้อมูลอย่างมีประสิทธิภาพ ฐานข้อมูลทำหน้าที่เป็นโครงสร้างพื้นฐานในการเก็บข้อมูลดิบ ขณะที่ระบบจัดการฐานข้อมูล (DBMS) ช่วยให้การจัดเก็บ การสืบค้น และการปรับปรุงข้อมูลเป็นไปอย่างรวดเร็วและมีประสิทธิภาพมากขึ้น การพัฒนาระบบที่ใช้แบบจำลองเชิงสัมพันธ์หรือ ระบบ NoSQL สำหรับข้อมูลที่มีโครงสร้างหลากหลายจึงเป็นสิ่งสำคัญต่อการตอบสนองความต้องการของธุรกิจยุคใหม่

การจัดการข้อมูลไม่เพียงแต่เกี่ยวข้องกับการจัดเก็บข้อมูลเท่านั้น แต่ยังรวมถึงการวิเคราะห์และการใช้งานข้อมูลเพื่อสร้างมูลค่าเพิ่มในรูปแบบของความรู้ การจัดการความรู้ในองค์กร เช่น การรวบรวมความรู้เชิงปริยายของพนักงานและการสร้างฐานความรู้ช่วยเพิ่มประสิทธิภาพในการทำงานและการตัดสินใจ เทคโนโลยีอย่าง AI และ Machine Learning ยังเสริมสร้างกระบวนการนี้ให้มีความแม่นยำและฉลาดมากยิ่งขึ้น โดยเฉพาะเมื่อองค์กรต้องจัดการกับข้อมูลขนาดใหญ่ที่ซับซ้อน

อย่างไรก็ตาม การบริหารจัดการข้อมูลยังมีความท้าทายในเรื่องของความปลอดภัย ความเป็นส่วนตัว และจริยธรรมในการใช้งาน การพัฒนาแนวทางการกำกับดูแลข้อมูลและการปฏิบัติตามกฎหมายที่เกี่ยวข้องจึงเป็นสิ่งสำคัญ การเข้าใจทฤษฎีและการปฏิบัติใน Database and Knowledge Management ไม่เพียงแต่ช่วยเพิ่มประสิทธิภาพในกระบวนการทำงาน แต่ยังเป็นการสร้างความยั่งยืนในระยะยาวให้กับองค์กรในโลกที่ขับเคลื่อนด้วยข้อมูลอย่างเต็มรูปแบบ

2.2 เทคโนโลยีที่เกี่ยวข้อง

2.2.1 การพัฒนา Web Application ในส่วนของการ Coding

การพัฒนาแอปพลิเคชันในปัจจุบันนั้นเราสามารถทำได้ในหลายวิธี และเทคโนโลยีที่ใช้ในการพัฒนามีข้อดีข้อเสียที่แตกต่างกันขึ้นอยู่กับวัตถุประสงค์ หรือเงื่อนไข ข้อจำกัดของแอปพลิเคชันที่เราต้องการพัฒนา ซึ่งบางเทคโนโลยีมีค่าใช้จ่ายสูงในการพัฒนา แต่มีประสิทธิภาพสูงมาก บางเทคโนโลยีมีความยืดยุ่นสูงในการพัฒนา และค่าใช้จ่ายไม่วสูงมาก แต่ประสิทธิภาพของแอปไม่สูงมาก

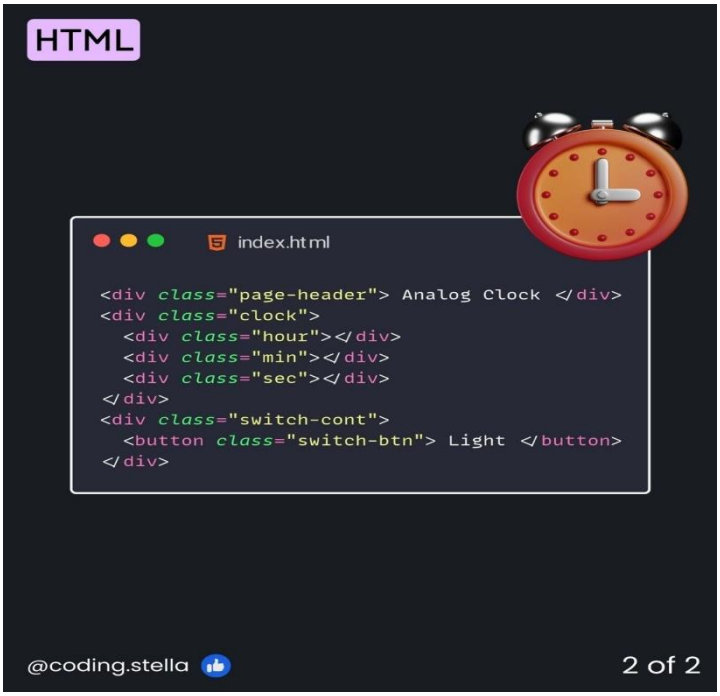
ในบทความ “**Web Programming คืออะไร ? ภาษาที่ต้องรู้มีอะไรบ้าง**” เขียนโดย **Mo Nuttamon**

6 ภาษาโปรแกรมหลัก ที่ใช้ทำ Web Programmingการทำ Web programming นั้นมีภาษาโปรแกรมมากมายที่สามารถใช้ได้ และแต่ละภาษาก็มีการใช้งานกับลักษณะที่ต่างกัน ตาม 6 ภาษาโปรแกรมหลัก ที่ Web programmer นิยมใช้กันต่อไปนี้

1. **HTML** ย่อมาจาก Hypertext Markup Language เป็นภาษาที่ใช้สำหรับสร้างโครงสร้างเว็บไซต์ โดยใช้ (Tag) ในการกำหนดส่วนต่าง ๆ ของเว็บอย่าง หัวข้อ ย่อหน้า ลิสต์รายการ ตาราง ภาพ ลิงก์HTML5 นั้นเป็นเวอร์ชันล่าสุดของ HTML ที่มีการพัฒนาเพิ่มคุณสมบัติใหม่เข้ามา เช่นการรองรับสื่อมัลติมีเดีย การสร้างฟอร์ม และการใช้ API ต่าง ๆ ถือเป็นภาษาพื้นฐานที่ Web programmer ทุกคนต้องรู้
2. **CSS** ย่อมาจาก Cascading Style Sheets คือภาษาที่ใช้กำหนดรูปแบบหรือสไตล์ของเว็บไซต์ โดยใช้ Selector, Property, Value และ Rule ในการกำหนด สี ขนาด ตำแหน่ง การจัดเรียง การปรับขนาด และอื่น ๆ ของ Element บนเว็บไซต์CSS3 นั้นเป็นเวอร์ชันล่าสุดของ CSS ที่เพิ่มคุณสมบัติใหม่เข้ามา อย่างการใช้สีแบบ Gradient การใช้ฟอนต์แบบ Web Font การใช้เอฟเฟกต์แบบ Transition/Transform และการจัดวางแบบ Flexbox, Grid ฯลฯ

เป็นภาษาโปรแกรมมิ่งสำคัญที่ใช้ออกแบบเว็บไซต์ให้สวยงาม และเหมาะสมกับขนาดหน้าจอของอุปกรณ์ที่แตกต่างกัน

3. **JavaScript** เป็นภาษาโปรแกรมมิ่งที่ใช้สำหรับเขียน Web programming ฝั่ง client-side เพื่อเพิ่มลูกเล่นโต้ตอบกับผู้ใช้ โดยทำงานผ่าน Web Browser เช่น Chrome Firefox หรือ Microsoft EdgeES6+ คือเวอร์ชันล่าสุดของ JavaScript ที่มีการเพิ่ม syntax ต่าง ๆ เพื่อช่วยให้การเขียนโค้ดทำได้ดี มีประสิทธิภาพ และใช้งานง่ายมากขึ้น
4. **TypeScript** ภาษาโปรแกรมที่เป็น Superset ของ JavaScript หรือหมายความว่า TypeScript จะมีคุณสมบัติเหมือน JavaScript แต่เพิ่มฟังก์ชันเพิ่มเติมเข้ามา อย่างการใช้ Type System ที่ช่วยให้การเขียนโค้ดมีความถูกต้องและง่ายต่อการอ่าน เพราะ TypeScript จะถูกคอมไพล์เป็น JavaScript ก่อนที่จะรันบนเว็บไซต์หรือเว็บแอปพลิเคชัน
5. **Python** ภาษาโปรแกรมที่เขียนเข้าใจง่ายและสามารถใช้ทำงานได้หลากหลาย เพราะมีไลบรารี (Library) กับเฟรมเวิร์ก (Framework) ให้เลือกใช้เยอะ อย่างเช่น Django, Flask, PyTorch หรือ TensorFlow สามารถใช้เขียน Web programming ได้ทั้งในส่วน Front end และ Back end



ภาพที่ 2.2.1 ตัวอย่างโค้ด HTML

อ้างอิงhttps://www.instagram.com/coding.stella/p/CzBZCtKP8Rm/?img_index=3

2.2.2 Cloud Services

ในการพัฒนาแอปพลิเคชันในปัจจุบัน การใช้บริการ Cloud Services นับเป็นหัวใจสำคัญที่ช่วยให้สามารถจัดการระบบได้อย่างสะดวกและมีประสิทธิภาพมากขึ้น โดยเฉพาะในด้านการจัดเก็บข้อมูลผู้ใช้งานที่มีจำนวนมาก ซึ่งการใช้บริการคลาวด์ทำให้ไม่จำเป็นต้องติดตั้งเซิร์ฟเวอร์และดูแลระบบเอง ช่วยลดต้นทุนทั้งในด้านฮาร์ดแวร์ บุคลากร และเวลา อีกทั้งยังสามารถขยายระบบ (scaling) ได้ง่ายเมื่อต้องรองรับผู้ใช้งานจำนวนมากขึ้น

สำหรับระบบที่พัฒนานี้ ได้เลือกใช้ MongoDB Atlas ซึ่งเป็นบริการฐานข้อมูลบนระบบคลาวด์ (Database-as-a-Service) จากบริษัท MongoDB Inc. โดย MongoDB Atlas ช่วยให้สามารถสร้าง จัดการ และดูแลฐานข้อมูล MongoDB ได้อย่างปลอดภัยผ่านแพลตฟอร์มคลาวด์ เช่น AWS, Google Cloud หรือ Azure โดยไม่ต้องติดตั้งและดูแลเซิร์ฟเวอร์เอง

MongoDB เป็นฐานข้อมูลแบบ NoSQL ที่เก็บข้อมูลในรูปแบบเอกสาร (document) ที่มีโครงสร้างคล้าย JSON ทำให้สามารถจัดเก็บข้อมูลที่มีความยืดหยุ่นและซับซ้อนได้ดี เหมาะกับแอปพลิเคชันยุคใหม่ที่มีการเปลี่ยนแปลงของข้อมูลอยู่ตลอดเวลา อีกทั้ง MongoDB Atlas ยังมีเครื่องมืออำนวยความสะดวกในการสำรองข้อมูล การติดตามสถานะ การตั้งค่าความปลอดภัย และการวิเคราะห์ข้อมูลอย่างครบถ้วน

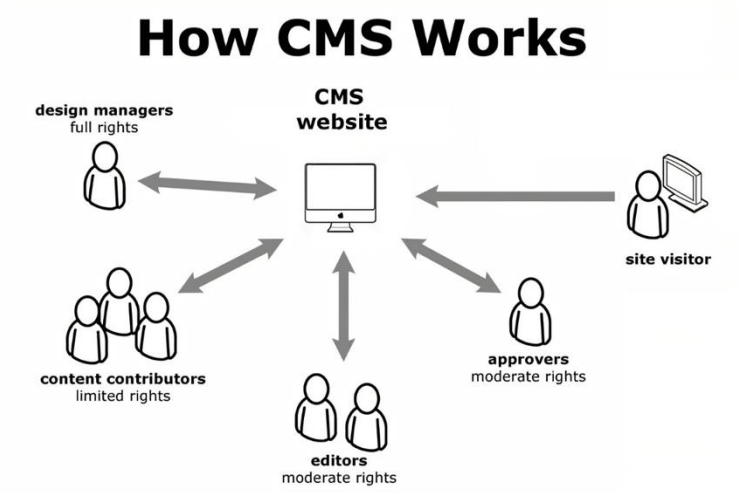
การเลือกใช้ MongoDB Atlas ช่วยให้สามารถโฟกัสกับการพัฒนาแอปพลิเคชันโดยไม่ต้องกังวลกับการดูแลโครงสร้างพื้นฐาน ส่งผลให้กระบวนการพัฒนาเป็นไปได้อย่างรวดเร็วและมีเสถียรภาพมากยิ่งขึ้น

นอกจากนี้ ในอนาคตระบบอาจมีการใช้เครื่องมืออื่น ๆ สำหรับการดีพลอย เช่น Netlify ซึ่งเป็นบริการคลาวด์สำหรับการนำเว็บแอปพลิเคชันขึ้นเผยแพร่ ที่รองรับการโหลดหน้าเว็บได้อย่างรวดเร็ว มีการจัดการโดเมนและการ deploy แบบต่อเนื่อง (Continuous Deployment) ช่วยให้การเผยแพร่แอปพลิเคชันเป็นเรื่องง่ายและทันสมัย

โดยอาจมี ตัว Deploy Netlify สำหรับการดีพลอยแอปพลิเคชันที่มีความเร็วในการโหลดสูง

2.2.3 CMS (Content Management System)

Content Management System (CMS) คือแพลตฟอร์มที่ช่วยให้ผู้ใช้งานสามารถสร้าง, แก้ไข, และจัดการเนื้อหาของเว็บไซต์หรือแอปพลิเคชันได้ง่าย โดยไม่ต้องมีความรู้ด้านการเขียนโค้ดมากมาย โดยเฉพาะในแอปพลิเคชันที่ต้องการให้ผู้ใช้สามารถสร้างเนื้อหาหรือโพสต์ได้เอง เช่น นิยายหรือบทความ



2.2.4 API

API ย่อมาจาก "Application Programming Interface" (อินเทอร์เฟซโปรแกรมประยุกต์) ในบริบทของ API คำว่า "Application" หมายถึงทุกซอฟต์แวร์ที่มีฟังก์ชันชัดเจน และ "Interface" อาจถือเป็นสัญญาบริการระหว่างแอปพลิเคชัน ซึ่งสัญญานี้จะกำหนดวิธีที่ทั้งสองฝ่ายสื่อสารกันโดยใช้คำขอและการตอบกลับ เอกสารประกอบ API มีข้อมูลเกี่ยวกับวิธีที่นักพัฒนาจัดโครงสร้างคำขอและการตอบกลับเหล่านั้น

API คือกลไกที่ช่วยให้ซอฟต์แวร์ของทั้งสองฝ่ายสามารถสื่อสารกันได้โดยใช้ชุดคำจำกัดความและโปรโตคอล ตัวอย่างเช่น ระบบซอฟต์แวร์ของนักพยากรณ์อากาศประกอบด้วยข้อมูลภาพอากาศรายวัน แอปพลิเคชันภาพอากาศในโทรศัพท์

ของคุณจะ "สื่อสาร" กับระบบนี้ผ่าน API และแสดงการอัปเดตภาพอากาศทุกวันบนโทรศัพท์ของคุณ

สถาปัตยกรรม API มักจะถูกอธิบายในแง่ของไคลเอ็นต์และเซิร์ฟเวอร์ แอปพลิเคชันที่ส่งคำขอเรียกว่าไคลเอ็นต์ และแอปพลิเคชันที่ส่งการตอบกลับเรียกว่าเซิร์ฟเวอร์ ในตัวอย่างภาพอากาศ ฐานข้อมูลภาพอากาศของสำนักงานคือเซิร์ฟเวอร์และแอปมือถือคือไคลเอ็นต์ โดย API ทำงานใน 4 รูปแบบด้วยกัน โดยขึ้นอยู่กับเวลาและสถานการณ์ที่สร้าง API

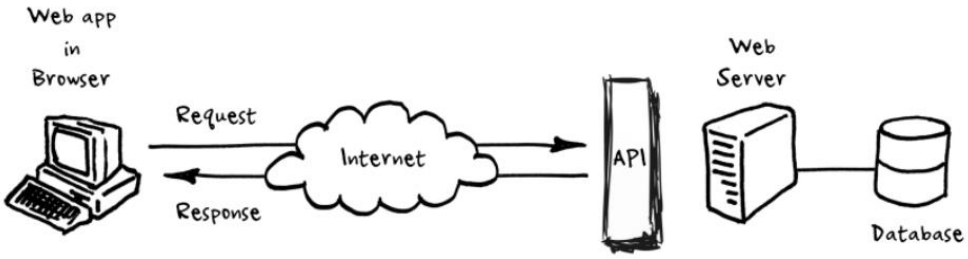
SOAP API เป็นมาตรฐานโปรโตคอลที่ใช้ในการสื่อสารระหว่างแอปพลิเคชันที่ต่างกันผ่านทางเครือข่าย โดยการสื่อสารนี้มีโครงสร้างข้อมูลที่ถูกกำหนดไว้ล่วงหน้าและใช้ภาษาที่เป็น XML (eXtensible Markup Language) ในการแลกเปลี่ยนข้อมูล SOAP API มักถูกนำมาใช้ในการเรียกเมธอด (methods) หรือการส่งข้อมูลระหว่างแอปพลิเคชันและเว็บเซิร์ฟเวอร์

RPC API หรือ Remote Procedure Call (การเรียกใช้กระบวนการระยะไกล) คือเทคโนโลยีที่ให้ความสามารถในการเรียกเมธอดหรือการดำเนินการที่ตั้งอยู่บนเซิร์ฟเวอร์จากระยะไกล ซึ่งทำให้แอปพลิเคชันสามารถเรียกใช้ฟังก์ชันหรือเมธอดบนเซิร์ฟเวอร์ได้โดยไม่ต้องรู้จักโค้ดภายในของเซิร์ฟเวอร์นั้นๆ และได้รับผลลัพธ์กลับมา ลักษณะของ RPC API มีลักษณะที่คล้ายกับการเรียกใช้ฟังก์ชันในโปรแกรมคอมพิวเตอร์ ซึ่งในกรณีนี้คือการเรียกใช้เมธอดหรือฟังก์ชันที่ตั้งอยู่บนเครื่องเซิร์ฟเวอร์จากระยะไกล โปรโตคอลที่ใช้ในการจัดการกับการเรียกเมธอดนี้มีรูปแบบที่มีโครงสร้างและเป็นมาตรฐาน เช่น XML-RPC หรือ JSON-RPC

WebSocket API เป็นเทคโนโลยีที่ใช้ในการเชื่อมต่อและสื่อสารระหว่างเบราว์เซอร์ (หรือแอปพลิเคชัน) กับเซิร์ฟเวอร์ในเวลาจริง โดยทำให้สามารถส่งข้อมูลไป-มาได้แบบครั้งเดียว (half-duplex) หรือแบบเต็ม (full-duplex) โดยที่การเชื่อมต่อถูกเรียกว่า "การเชื่อมต่อ WebSocket" (WebSocket connection) WebSocket API นำเสนอแนวคิดที่แตกต่างกับโปรโตคอล HTTP ที่ใช้สำหรับการ

สื่อสารในโลกเว็บ ในโปรโตคอล HTTP เมื่อเบราว์เซอร์ต้องการข้อมูลใหม่ มันจะต้องส่งคำร้องขอ (request) ไปที่เซิร์ฟเวอร์และรับคำตอบ (response) จากนั้นปิดการเชื่อมต่อ แต่ WebSocket สามารถเปิดการเชื่อมต่อไว้ตลอดเวลา ทำให้สามารถส่งข้อมูลไป-มาได้โดยไม่ต้องสร้างการเชื่อมต่อใหม่ทุกครั้ง

REST API ย่อมาจาก Representational State Transfer (การโอนสถานะแบบแทนที่) REST ช่วยกำหนดชุดฟังก์ชันต่างๆ เช่น GET, PUT, DELETE ฯลฯ ที่ไคลเอ็นต์สามารถใช้เพื่อเข้าถึงข้อมูลเซิร์ฟเวอร์ได้ ไคลเอ็นต์และเซิร์ฟเวอร์แลกเปลี่ยนข้อมูลโดยใช้ HTTP คุณสมบัติหลักของ REST API คือ ความเป็นอิสระ ความเป็นอิสระหมายความว่าเซิร์ฟเวอร์จะไม่บันทึกข้อมูลไคลเอ็นต์ระหว่างคำขอ คำขอของไคลเอ็นต์ไปยังเซิร์ฟเวอร์นั้นคล้ายกับ URL ที่คุณพิมพ์ในเบราว์เซอร์ของคุณเพื่อเยี่ยมชมเว็บไซต์ การตอบสนองจากเซิร์ฟเวอร์เป็นข้อมูลธรรมดา โดยไม่มีการแสดงผลแบบกราฟิกทั่วไปของหน้าเว็บ



2.2.5 ระบบพจนานุกรม

2.2.5.1 Python 3.x คือเวอร์ชันของภาษา Python ที่ได้รับการพัฒนาให้ทันสมัยและเหมาะกับการเขียนโปรแกรมในยุคใหม่ โดยมีจุดเด่นในด้านความอ่านง่าย โครงสร้างภาษาชัดเจน และมีไลบรารีสนับสนุนจำนวนมาก Python เป็นภาษาที่ได้รับความนิยมทั่วโลก และเวอร์ชัน 3.x ถือเป็นมาตรฐานที่ใช้กันอย่างแพร่หลายในวงการพัฒนาเว็บ, วิทยาศาสตร์ข้อมูล, ปัญญาประดิษฐ์ และการประมวลผลภาษาธรรมชาติ

ในระบบพจนานุกรมนี้ Python 3.x ถูกนำมาใช้เป็นภาษาหลักในการพัฒนา โดยใช้ร่วมกับไลบรารีอื่น ๆ เช่น Flask, PyThaiNLP และ nltk เพื่อสร้าง API ที่รับคำศัพท์จากผู้ใช้ ประมวลผลความหมาย และส่งผลลัพธ์กลับในรูปแบบที่เข้าใจง่าย นอกจากนี้ Python ยังช่วยให้การพัฒนาและทดสอบระบบทำได้รวดเร็วและมีประสิทธิภาพ

2.2.5.2 Flask คือเฟรมเวิร์กสำหรับพัฒนาเว็บแอปพลิเคชันด้วยภาษา Python ซึ่งออกแบบมาให้มีขนาดเล็ก ใช้งานง่าย และยืดหยุ่นสูง Flask เหมาะสำหรับการสร้าง RESTful API ที่สามารถรับข้อมูลจากผู้ใช้ ประมวลผล และตอบกลับผลลัพธ์ได้อย่างรวดเร็ว โดยมีจุดเด่นคือสามารถเริ่มต้นใช้งานได้ทันทีโดยไม่ต้องกำหนดโครงสร้างโครงการที่ซับซ้อน

ในระบบพจนานุกรม Flask ถูกใช้เป็นแกนหลักในการสร้าง Web API สำหรับรับคำศัพท์จากผู้ใช้ผ่าน HTTP request แล้วส่งผลลัพธ์กลับในรูปแบบ JSON เพื่อให้สามารถเชื่อมต่อกับฝั่ง Frontend ที่พัฒนาด้วย React ได้อย่างไร้รอยต่อ

2.2.5.3 PyThaiNLP เป็นไลบรารีโอเพนซอร์สที่ใช้สำหรับประมวลผลภาษาธรรมชาติ (NLP) ของภาษาไทย โดยสนับสนุนความสามารถต่าง ๆ เช่น ตัดคำ การแปลงเสียง การสะกดคำ และการเข้าถึงฐานข้อมูล WordNet ภาษาไทย

ในระบบพจนานุกรม PyThaiNLP ถูกใช้สำหรับเรียกข้อมูลความหมายของคำศัพท์จากคลัง WordNet ซึ่งเป็นฐานข้อมูลคำศัพท์แบบมีโครงสร้าง (lexical database) ทำให้สามารถแสดงความหมายของคำในเชิงนิยามได้อย่างถูกต้องและแม่นยำ

2.2.5.4 Google Translate API คือบริการแปลภาษาของ Google ซึ่งในโปรเจกต์นี้มีการใช้เวอร์ชันฟรีผ่าน URL translate.googleapis.com โดยไม่ต้องใช้ API Key การใช้งานแบบนี้เหมาะสำหรับโปรเจกต์ทดสอบหรือระบบที่ไม่ได้เรียกใช้ในปริมาณมาก

ในระบบพจนานุกรม Google Translate ถูกนำมาใช้เพื่อแปลคำศัพท์จากภาษาไทยเป็นอังกฤษ หรืออังกฤษเป็นไทย เพื่อช่วยให้ผู้ใช้เข้าใจคำศัพท์ในอีกภาษาหนึ่งควบคู่กับคำนิยามที่แสดงจาก WordNet

2.2.5.5 Natural Language Toolkit (nltk) คือไลบรารียอดนิยมของ Python ที่ใช้สำหรับการประมวลผลภาษาธรรมชาติ มีเครื่องมือและคลังข้อมูลหลากหลายชนิด เช่น WordNet, stopwords และ corpus อื่น ๆ ที่สามารถใช้งานได้ฟรี

ในระบบพจนานุกรม nltk ทำหน้าที่เป็นตัวกลางให้ PyThaiNLP เข้าถึงข้อมูล WordNet โดยเฉพาะส่วนประกอบสำคัญอย่าง omw-1.4 ที่ใช้สำหรับความหมายข้ามภาษา ทำให้สามารถดึงนิยามของคำศัพท์มาแสดงได้แม่นยำและรวดเร็ว

2.2.5.6 flask-cors คือส่วนเสริมของ Flask ที่ช่วยให้ Web API สามารถรับการเชื่อมต่อจาก frontend ที่รันอยู่คนละโดเมนหรือคนละพอร์ต (Cross-Origin Resource Sharing - CORS) ได้ เช่น React frontend ที่รันอยู่ที่ localhost:3000 จะสามารถส่งคำขอไปยัง Flask backend ที่รันที่ localhost:5000

การใช้ flask-cors จึงมีความสำคัญอย่างยิ่งในการทำให้ระบบสามารถทำงานร่วมกันได้อย่างสมบูรณ์ โดยเฉพาะในระหว่างการพัฒนาแบบแยกฝั่ง frontend และ backend

บทที่ 3
วิธีการวิจัย

3.1 ภาพรวมของโครงงาน

ในการพัฒนาเว็ปแอปพลิเคชัน Taledge หรือ แอปพลิเคชันสำหรับนักเขียนนิยาย นั้น มีขั้นตอนและวิธีการดำเนินงานดังนี้

3.1.1 แนวคิด และ software architecture

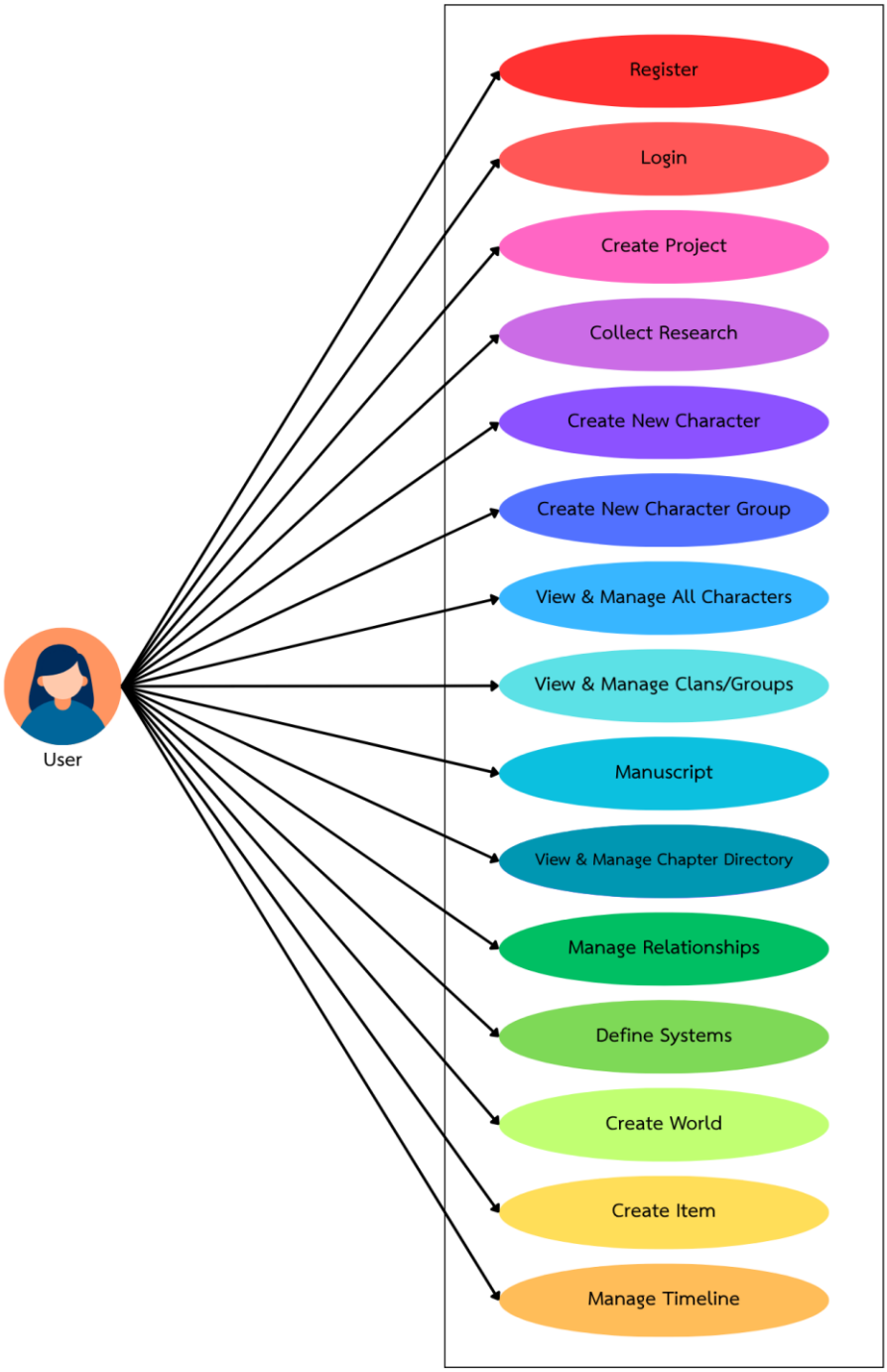
โดยเว็ปแอปพริเคชัน Taledge เริ่มต้นจากการวิเคราะห์ปัญหาและความต้องการของนักเขียนนิยาย โดยเน้นไปที่ความท้าทายในการจัดการพล็อตเรื่อง ตัวละคร และไทม์ไลน์ เพื่อให้ผู้เขียนสามารถจัดระเบียบและพัฒนางานเขียนได้อย่างมีประสิทธิภาพ จากนั้นได้กำหนดวัตถุประสงค์และขอบเขตของระบบ พร้อมระบุฟังก์ชันหลัก เช่น การสร้างพล็อต การจัดการข้อมูลตัวละคร และการแสดงความสัมพันธ์ระหว่างองค์ประกอบในนิยาย เพื่อรองรับการใช้งานที่หลากหลายและตอบโจทย์ผู้เขียนนิยายทุกระดับ

ในส่วนการออกแบบระบบ จะเริ่มจากการวางโครงสร้างฐานข้อมูลเชิงสัมพันธ์เพื่อจัดเก็บข้อมูล เช่น ตัวละคร เหตุการณ์ และไทม์ไลน์ รวมถึงการออกแบบ UX/UI เพื่อสร้างอินเทอร์เฟซที่ใช้งานง่ายและตอบสนองต่อความต้องการของผู้ใช้งาน พร้อมทั้งการจัดทำแผนภาพ Use Case เพื่ออธิบายกระบวนการทำงานของระบบในแต่ละฟังก์ชัน หลังจากนั้นได้พัฒนาแอปพลิเคชันโดยใช้เทคโนโลยีต่าง ๆ เช่น HTML, CSS, JavaScript ในส่วน Front-end และการใช้บริการ Cloud Services เช่น AWS ในการจัดเก็บข้อมูลและโฮสต์ระบบ

เพื่อที่จะพัฒนาระบบให้เสร็จสมบูรณ์ จะต้องมีการทดสอบการทำงานในทุกฟังก์ชัน เช่น การสร้างและบันทึกพล็อต รวมถึงการแสดงความสัมพันธ์ระหว่างตัวละครและไทม์ไลน์ เพื่อตรวจสอบความถูกต้องและปรับปรุงข้อผิดพลาด และสุดท้ายคือการจัดทำรายงานและสรุปผลการดำเนินงาน

3.2 การวิเคราะห์ขอบเขตและความต้องการของระบบ

สิ่งที่ระบบต้องการเพื่อตอบสนองผู้ใช้ มีดังนี้



3.2.1 Functional Requirement

3.2.1.1 Use case

Use case ID	UC-001
Use case name	Register User (การลงทะเบียนผู้ใช้งาน)
Description	ผู้ใช้งานทั่วไปสามารถสร้างบัญชีใหม่เพื่อเข้าถึงฟีเจอร์ต่างๆ
Trigger	ผู้ใช้งานคลิกปุ่ม "สมัครสมาชิก"
Actor	ผู้ใช้งานทั่วไป (General User)
Preconditions	ผู้ใช้งานต้องเข้าถึงหน้า "สมัครสมาชิก" บนเว็บไซต์
Post-conditions	บัญชีผู้ใช้งานถูกสร้างและบันทึกในระบบสำเร็จ
Basic flow	1. ผู้ใช้กดปุ่ม "สมัครสมาชิก" 2. กรอกข้อมูล เช่น ชื่อผู้ใช้ อีเมล และรหัสผ่าน 3. กดปุ่ม "ยืนยัน" 4. บัญชีถูกเปิดใช้งาน
Alternative flow	A1: ข้อมูลไม่ครบถ้วน 1. ระบบตรวจพบว่าผู้ใช้กรอกข้อมูลไม่ครบ 2. ระบบแสดงข้อความแจ้งเตือนและให้ผู้ใช้กรอกข้อมูลใหม่ 3. ผู้ใช้กรอกข้อมูลให้ครบและกด "ยืนยัน" A2: อีเมลซ้ำในระบบ

	1. ระบบตรวจพบว่าอีเมลที่กรอกอยู่ในระบบแล้ว 2. ระบบแจ้งเตือนว่าอีเมลนี้ถูกใช้งานไปแล้ว 3. ผู้ใช้สามารถเลือกเปลี่ยนอีเมลหรือกู้คืนบัญชี

Use case ID	UC-002
Use case name	Login (เข้าสู่ระบบ)
Description	สมาชิกสามารถเข้าสู่ระบบเพื่อใช้งานฟีเจอร์ต่างๆ
Trigger	ผู้ใช้คลิกปุ่ม "เข้าสู่ระบบ"
Actor	สมาชิก (Registered User)
Preconditions	ผู้ใช้ต้องมีบัญชีที่ลงทะเบียนไว้
Post-conditions	ผู้ใช้สามารถเข้าสู่ระบบและเข้าถึงแดชบอร์ดส่วนตัว
Basic flow	1. ผู้ใช้กดปุ่ม "เข้าสู่ระบบ" 2. กรอกชื่อผู้ใช้และรหัสผ่าน 3. กดปุ่ม "ยืนยัน" 4. ระบบตรวจสอบข้อมูลและพาผู้ใช้เข้าสู่แดชบอร์ด
Alternative flow	A1: รหัสผ่านผิด 1. ระบบตรวจพบว่ารหัสผ่านไม่ถูกต้อง 2. ระบบแสดงข้อความแจ้งเตือน 3. ผู้ใช้กรอกรหัสผ่านใหม่

Use case ID	UC-003
Use case name	Create Project (สร้างโปรเจกต์งานเขียน)
Description	สมาชิกสามารถสร้างโปรเจกต์ใหม่เพื่อเริ่มการสร้างสรรค์งานเขียน
Trigger	กดปุ่ม "สร้างโปรเจกต์ใหม่"
Actor	สมาชิก (Registered User)
Preconditions	ผู้ใช้ต้องเข้าสู่ระบบสำเร็จ
Post-conditions	โปรเจกต์ใหม่ถูกบันทึกในระบบ
Basic flow	1. ผู้ใช้เข้าสู่แดชบอร์ด 2. กดปุ่ม "สร้างโปรเจกต์ใหม่" 3. ใส่ข้อมูลโปรเจกต์ เช่น ชื่อโปรเจกต์ คำโปรย 4. กดปุ่ม "บันทึก" 5. ระบบแสดงโปรเจกต์ใหม่ในรายการ
Alternative flow	A1: ชื่อโปรเจกต์ซ้ำ 1. ระบบแจ้งเตือนว่าชื่อโปรเจกต์นี้ถูกใช้งานแล้ว 2. ผู้ใช้แก้ไขชื่อโปรเจกต์ใหม่ 3. กดปุ่ม "บันทึก" อีกครั้ง

Use case ID	UC-004
Use case name	Research (จัดเก็บข้อมูลอ้างอิงหรือไอเดีย)
Description	ผู้ใช้สามารถเพิ่มข้อมูลอ้างอิง แหล่งข้อมูล รูปภาพ วิดีโอ ไฟล์ PDF หรือไอเดียอื่น ๆ เพื่อประกอบการแต่งนิยาย
Trigger	ผู้ใช้คลิกที่โปรเจกต์และเลือกเมนู Research จาก sidebar
Actor	สมาชิก (Registered User)
Preconditions	ต้องมีโปรเจกต์ที่สร้างไว้แล้ว
Post-conditions	ข้อมูล Research ถูกบันทึกหรืออัปเดตในระบบ
Basic flow	1. ผู้ใช้เลือกเมนู "Research" 2. ระบบแสดง Research Panel 3. ผู้ใช้กด “+ Add Panel” 4. ผู้ใช้เลือกประเภท (ข้อความ, รูปภาพ, ลิงก์, PDF, วิดีโอ ฯลฯ) 5. กรอกข้อมูลที่ต้องการ เช่น title, description, แนบไฟล์ 6. กด “บันทึก” 7. ระบบแสดง panel ใหม่ในหน้า Research
Alternative flow	A1: ผู้ใช้กด “ยกเลิก” 1. ระบบยกเลิกและไม่บันทึกข้อมูล

Use case ID	UC-005
Use case name	New Character (สร้างตัวละครใหม่)
Description	สมาชิกสามารถสร้างตัวละครพร้อมระบุชื่อ ลักษณะ และข้อมูลอื่นๆ
Trigger	ผู้ใช้เลือกเมนู "New Character"
Actor	สมาชิก (Registered User)
Preconditions	ต้องมีโปรเจกต์ที่สร้างไว้แล้ว
Post-conditions	ตัวละครใหม่ถูกบันทึกในระบบ
Basic flow	1. ผู้ใช้กดปุ่ม “New Character” 2. ระบบเปิดแบบฟอร์มสำหรับสร้างตัวละคร 3. ผู้ใช้กรอกข้อมูล: ชื่อ, คำอธิบาย, เพศ, กลุ่ม, ความสามารถ ฯลฯ 4. กดปุ่ม “บันทึก” 5. ระบบบันทึกข้อมูลและแสดงตัวละครใหม่ใน All Characters
Alternative flow	A1: ไม่กรอกชื่อ 1. ระบบแจ้งเตือนให้กรอกชื่อ 2. ผู้ใช้กรอกชื่อและกดบันทึกใหม่

Use case ID	UC-006
Use case name	New Character Group (สร้างกลุ่มตัวละคร เช่น ตระกูล/องค์กร)
Description	สมาชิกสามารถสร้างกลุ่มเพื่อรวบรวมตัวละครที่เกี่ยวข้องกัน
Trigger	ผู้ใช้เลือกเมนู “New Character Group”
Actor	สมาชิก (Registered User)
Preconditions	ต้องมีโปรเจกต์ที่สร้างไว้แล้ว
Post-conditions	กลุ่มใหม่ถูกสร้างและบันทึกในระบบ
Basic flow	1. ผู้ใช้เลือก "New Character Group" 2. กรอกชื่อกลุ่มและรายละเอียดอื่นๆ 3. กด "บันทึก"
Alternative flow	A1: ไม่กรอกชื่อ 1. ระบบแจ้งเตือนให้กรอกชื่อ 2. ผู้ใช้กรอกชื่อและกดบันทึกใหม่

Use case ID	UC-007
Use case name	All Characters (ดูและจัดการตัวละครทั้งหมด)
Description	สมาชิกสามารถดู แก้ไข หรือลบตัวละครที่เคยสร้างไว้
Trigger	ผู้ใช้เลือกเมนู “All Characters”
Actor	สมาชิก (Registered User)
Preconditions	ต้องมีโปรเจกต์ที่สร้างไว้แล้ว
Post-conditions	ข้อมูลตัวละครถูกอัปเดตหรือลบ
Basic flow	1. ผู้ใช้คลิกเมนู "All Characters" 2. ระบบแสดงรายการตัวละครทั้งหมด 3. ผู้ใช้เลือกตัวละครเพื่อดูรายละเอียด 4. กด “แก้ไข” เพื่อเปลี่ยนแปลงข้อมูล แล้วกด “บันทึก” 5. หรือ กด “ลบ” และยืนยันการลบ 6. ระบบอัปเดตรายการตัวละคร
Alternative flow	-

Use case ID	UC-008
Use case name	Clans (ดูและจัดการกลุ่มตัวละคร)
Description	สมาชิกสามารถดู แก้ไข หรือลบกลุ่มตัวละครที่สร้างไว้
Trigger	ผู้ใช้เลือกเมนู “Clans”
Actor	สมาชิก (Registered User)
Preconditions	ต้องมีโปรเจกต์ที่สร้างไว้แล้ว และต้องมีอย่างน้อย 1 กลุ่มที่สร้างไว้
Post-conditions	ข้อมูลกลุ่มถูกอัปเดตหรือลบ
Basic flow	1. ผู้ใช้เลือก "Clans" 2. เลือกกลุ่มที่ต้องการดู 3. แก้ไขรายละเอียด หรือลบ 4. ระบบอัปเดตข้อมูล
Alternative flow	-

Use case ID	UC-009
Use case name	Manuscript (การเขียนต้นฉบับ)
Description	สมาชิกสามารถเขียนและแก้ไขงานเขียนต้นฉบับของโปรเจกต์ได้
Trigger	ผู้ใช้เลือกเมนู "Manuscript" จากโปรเจกต์
Actor	สมาชิก (Registered User)
Preconditions	ต้องมีโปรเจกต์ที่สร้างไว้แล้ว
Post-conditions	เนื้อหาต้นฉบับถูกบันทึกในระบบ
Basic flow	1. ผู้ใช้เข้าสู่โปรเจกต์และเลือกเมนู "Manuscript" 2. เริ่มพิมพ์หรือแก้ไขเนื้อหาในพื้นที่เขียน 3. ระบบบันทึกเนื้อหาต้นฉบับ
Alternative flow	-

Use case ID	UC-010
Use case name	Chapter Directory (จัดการต้นฉบับทั้งหมดที่เคยเขียน)
Description	สมาชิกสามารถดู แก้ไข หรือลบต้นฉบับแต่ละตอนที่เคยเขียนไว้
Trigger	ผู้ใช้เลือกเมนู “Chapter Directory”
Actor	สมาชิก (Registered User)
Preconditions	ต้องมีโปรเจกต์ที่สร้างไว้แล้ว และต้องมีต้นฉบับที่เคยเขียนไว้อย่างน้อย 1 ตอน
Post-conditions	ข้อมูลบทถูกอัปเดตหรือลบ
Basic flow	1. ผู้ใช้คลิกเมนู “Chapter Directory” 2. ระบบแสดงรายการต้นฉบับ 3. ผู้ใช้เลือก chapter เพื่ออ่านหรือแก้ไข 4. ลากบทไปยังพื้นที่ “ลบ” เพื่อลบ chapter (มีการยืนยัน)
Alternative flow	-

Use case ID	UC-011
Use case name	Relationships (จัดการความสัมพันธ์ตัวละคร)
Description	สมาชิกสามารถกำหนดความสัมพันธ์ระหว่างตัวละคร เช่น พี่น้อง ศัตรู คนรัก ฯลฯ
Trigger	ผู้ใช้เลือกเมนู “Relationships”
Actor	สมาชิก (Registered User)
Preconditions	ต้องมีโปรเจกต์ที่สร้างไว้แล้ว และมีตัวละครอย่างน้อย 2 ตัวในโปรเจกต์
Post-conditions	ความสัมพันธ์ถูกบันทึก
Basic flow	1. ผู้ใช้เลือก "Relationships" 2. ระบบแสดงตัวละครทั้งหมด 3. ผู้ใช้เลือก 2 ตัวละคร 4. ใส่รายละเอียดความสัมพันธ์ (พี่น้อง, ศัตรู ฯลฯ) 5. กด “บันทึก”
Alternative flow	-

Use case ID	UC-012
Use case name	Systems (สร้างระบบ เช่น ชนชั้น, ระดับพลัง, ฯลฯ)
Description	สมาชิกสามารถสร้างและจัดการระบบต่างๆ ของโลกในนิยาย เช่น ระบบเวทมนตร์ ระบบเศรษฐกิจ หรือระบบการปกครอง
Trigger	ผู้ใช้เลือกเมนู “Systems”
Actor	สมาชิก (Registered User)
Preconditions	ต้องมีโปรเจกต์ที่สร้างไว้แล้ว
Post-conditions	ระบบที่สร้างถูกบันทึก
Basic flow	1. ผู้ใช้คลิกที่เมนู “Systems” 2. ผู้ใช้กด “+ Root” 3. กรอกชื่อระบบ และรายละเอียด 4. กดปุ่ม “บันทึก”
Alternative flow	-

Use case ID	UC-013
Use case name	World (สร้างโลกของเรื่องราว)
Description	สมาชิกสามารถสร้างโลกของนิยาย พร้อมระบุลักษณะทางภูมิศาสตร์ ประวัติศาสตร์ และองค์ประกอบต่างๆ
Trigger	ผู้ใช้เลือกเมนู “World”
Actor	สมาชิก (Registered User)
Preconditions	ต้องมีโปรเจกต์ที่สร้างไว้แล้ว
Post-conditions	โลกนิยายถูกบันทึก
Basic flow	1. ผู้ใช้คลิกเมนู “World” 2. ระบบแสดงรายการโลกที่เคยสร้าง (ถ้ามี) และปุ่ม “+ Add Panel” 3. ผู้ใช้กด “+ Add Panel” 4. กรอกชื่อโลก, คำอธิบาย, ประวัติ, แนบแผนที่/รูป (ถ้ามี) 5. กด “บันทึก” 6. ระบบแสดงโลกใหม่ในรายการ
Alternative flow	A1: ไม่กรอกชื่อ 1. ระบบแจ้งเตือนให้กรอกชื่อ 2. ผู้ใช้กรอกชื่อและกดบันทึกใหม่ A2: ผู้ใช้กด “ยกเลิก” 1. ระบบยกเลิกและไม่บันทึกข้อมูล

Use case ID	UC-014
Use case name	Item (สร้างไอเทมในเรื่อง)
Description	สมาชิกสามารถสร้างไอเทมหรือสิ่งของ เช่น อาวุธ เครื่องมือ เครื่องราง ฯลฯ เพื่อใช้ในเรื่อง
Trigger	ผู้ใช้เลือกเมนู “Items”
Actor	สมาชิก (Registered User)
Preconditions	ต้องมีโปรเจกต์ที่สร้างไว้แล้ว
Post-conditions	ไอเทมถูกบันทึกลงระบบ
Basic flow	1. ผู้ใช้คลิกเมนู “Item” 2. ระบบแสดงรายการไอเทมที่เคยสร้าง (ถ้ามี) และปุ่ม “+ Add Panel” 3. ผู้ใช้กด “+ Add Panel” 4. กรอกชื่อ, คำอธิบาย, คุณสมบัติ, แนบรูปภาพ (ถ้ามี) 5. กด “บันทึก” 6. ระบบแสดงไอเทมใหม่ในรายการ
Alternative flow	A1: ไม่กรอกชื่อ 3. ระบบแจ้งเตือนให้กรอกชื่อ 4. ผู้ใช้กรอกชื่อและกดบันทึกใหม่ A2: ผู้ใช้กด “ยกเลิก” 1. ระบบยกเลิกและไม่บันทึกข้อมูล

Use case ID	UC-015
Use case name	Timeline (สร้างและจัดการเส้นเวลาเรื่องราว)
Description	สมาชิกสามารถเพิ่ม แก้ไข หรือลบเหตุการณ์ในไทม์ไลน์ของเรื่องราว
Trigger	ผู้ใช้เลือกเมนู “Timeline”
Actor	สมาชิก (Registered User)
Preconditions	ต้องมีโปรเจกต์ที่สร้างไว้แล้ว
Post-conditions	เหตุการณ์ต่างๆ ถูกบันทึกลงระบบ
Basic flow	1. ผู้ใช้เลือก "Timeline" 2. กด “+ New” กรอกชื่อ Timeline ที่ต้องการสร้าง 3. เพิ่มเหตุการณ์ใหม่พร้อมวันเวลาและคำอธิบาย 4. กด "บันทึก"
Alternative flow	A1: ไม่กรอกชื่อเหตุการณ์และวันที่ 1. ระบบแจ้งเตือนให้กรอกชื่อและวันที่ 2. ผู้ใช้กรอกชื่อและวันที่ แล้วกดบันทึกใหม่

3.2.2 Non-Functional Requirement

1. **อินเทอร์เฟซที่ใช้งานง่าย (Usability)** ระบบมีอินเทอร์เฟซที่นักเขียนสามารถเข้าใจและใช้งานได้ง่าย โดยไม่ต้องมีการฝึกอบรม
2. **การใช้งานข้ามแพลตฟอร์ม (Portability)** รองรับการใช้งานบนอุปกรณ์ต่าง ๆ เช่น คอมพิวเตอร์ สมาร์ทโฟน และแท็บเล็ต
3. **ความปลอดภัยของข้อมูล (Security)** มีระบบจัดเก็บข้อมูลที่ปลอดภัยและการสำรองข้อมูลอัตโนมัติ
4. **ประสิทธิภาพในการโหลดข้อมูล (Performance)** ระบบสามารถทำงานได้ราบรื่นและตอบสนองได้รวดเร็วแม้มีข้อมูลจำนวนมาก
5. **การรองรับไฟล์หลากหลายรูปแบบ (Interoperability)** รองรับการนำเข้าและส่งออกไฟล์ในรูปแบบมาตรฐาน เช่น .docx, .pdf, และ .epub

3.3 การออกแบบระบบ

3.3.1 Architecture Overview

ระบบ TALEDGE ออกแบบในลักษณะ Client-Server โดยมีการแยกส่วน Frontend และ Backend อย่างชัดเจน โดยมีการใช้ RESTful API เป็นตัวกลางในการเชื่อมต่อข้อมูลระหว่างสองฝั่ง ระบบสามารถสรุปได้ดังนี้:

- **Client (Frontend):** React.js
- **Server (Backend):** Node.js + Express.js
- **Database:** MongoDB Atlas
- **Hosting:** Netlify (Frontend) และ Render (Backend)

เทคโนโลยีที่ใช้

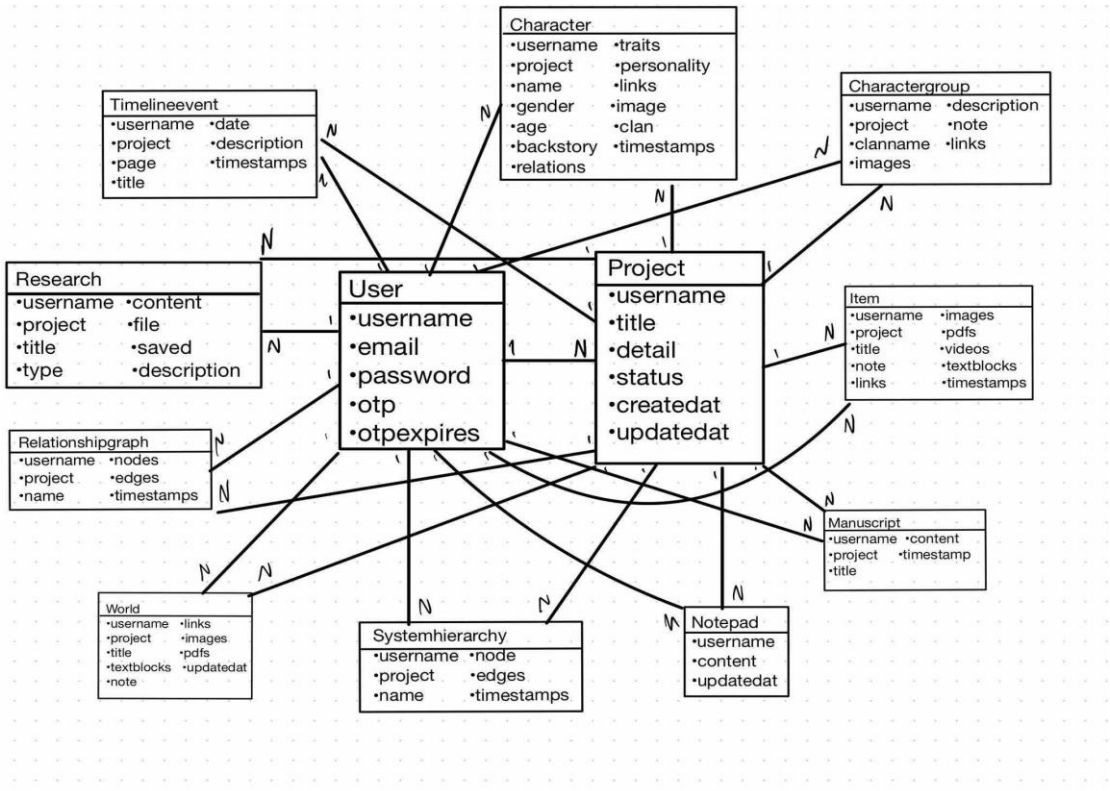
หมวดหมู่	เทคโนโลยี
Frontend	HTML5, CSS3, JavaScript (ES6), React.js
Backend	Node.js, Express.js, Python (Flask)
Database	MongoDB Atlas (NoSQL)
API	RESTful API, Flask API (สำหรับพจนานุกรม)
Tools	Git, VS Code, Postman

3.3.2 การออกแบบฐานข้อมูล

ระบบใช้ฐานข้อมูล NoSQL โดยออกแบบเป็น Document-Based Structure มี Schema หลักๆ ได้แก่:

- User: ข้อมูลผู้ใช้งาน เช่น username, email, password
- Project: ชื่อโปรเจกต์และ metadata
- Character, Event, Item, World, Research, System: องค์ประกอบที่เกี่ยวข้องกับโครงเรื่อง

ER-Diagram ของระบบ Taledge

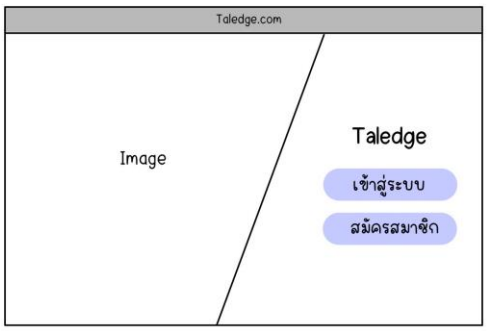


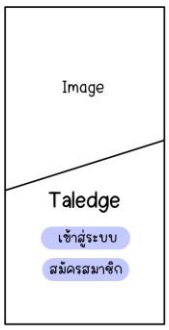
3.3.3 การออกแบบส่วนต่อประสาน (UI/UX)

อินเทอร์เฟซของระบบมี Sidebar สำหรับเลือก Section ต่าง ๆ เช่น Timeline, Characters, Manuscript เป็นต้น

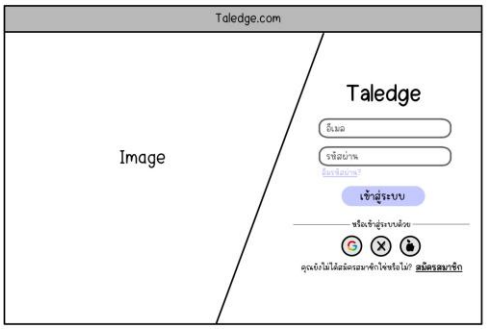
ภาพ Mockup ประกอบ:

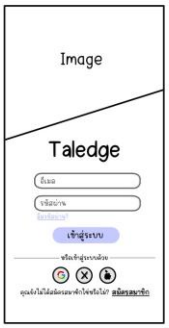
หน้าเริ่มต้น



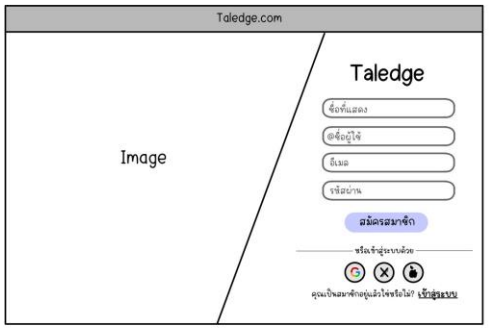


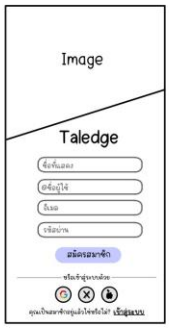
หน้าเข้าสู่ระบบ



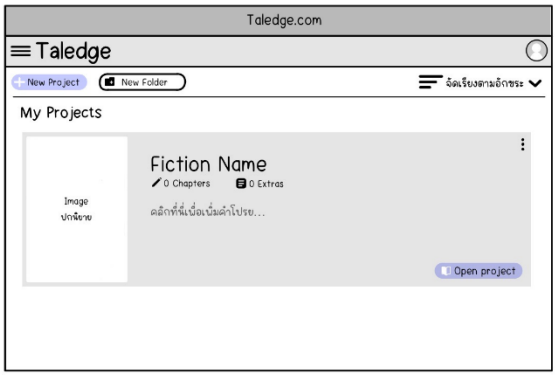


หน้าสมัครสมาชิก

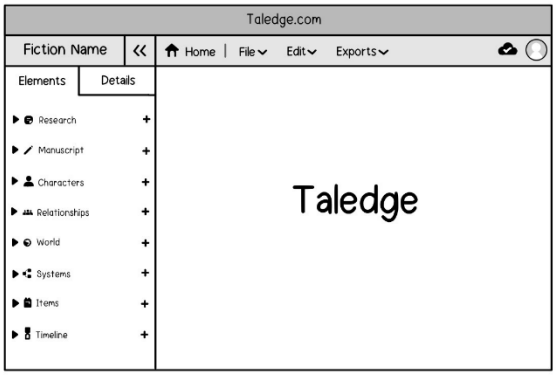




หน้า Home/แดชบอร์ดส่วนตัว

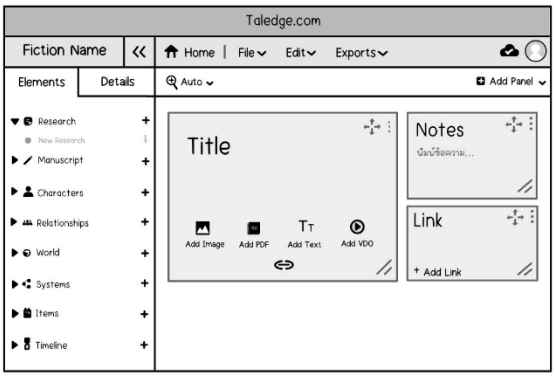


หน้าเริ่มต้น Project

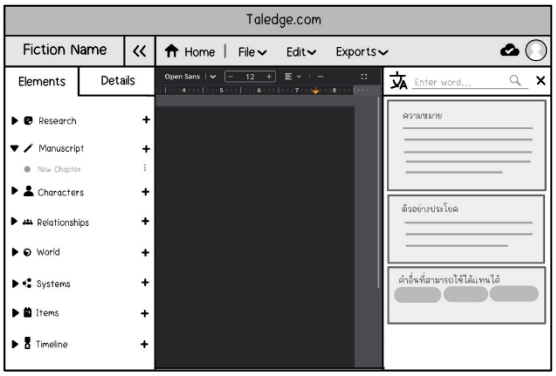


หน้า Research

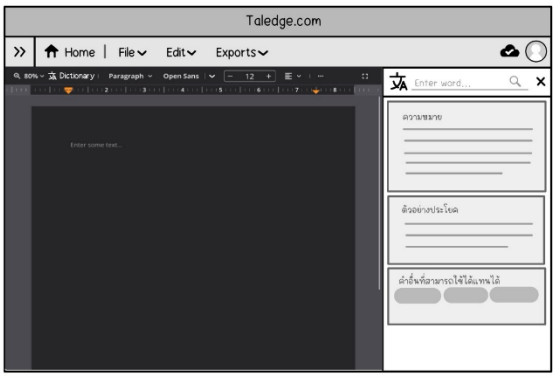
สำหรับเก็บไอเดียต่างๆ ข้อมูลที่เอามาใช้ในนิยาย



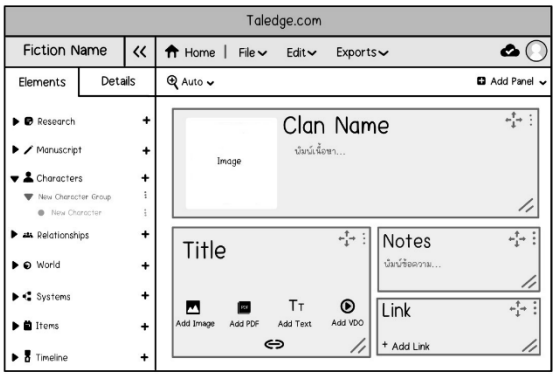
หน้าพิมพ์นิยาย



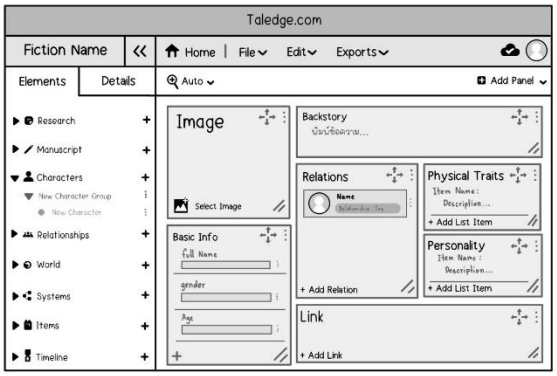
ขับจอ elements



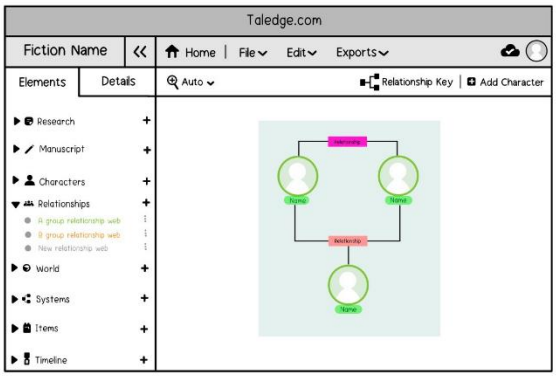
หน้า Characters Group Character



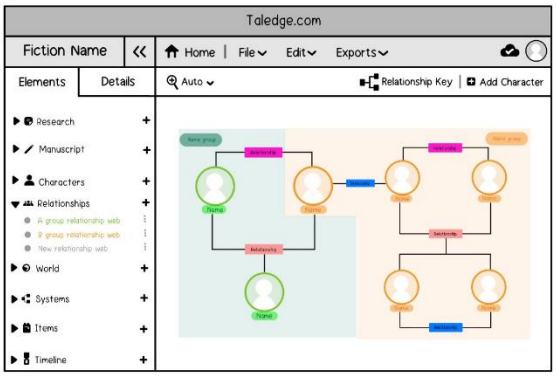
หน้า Characters New Character



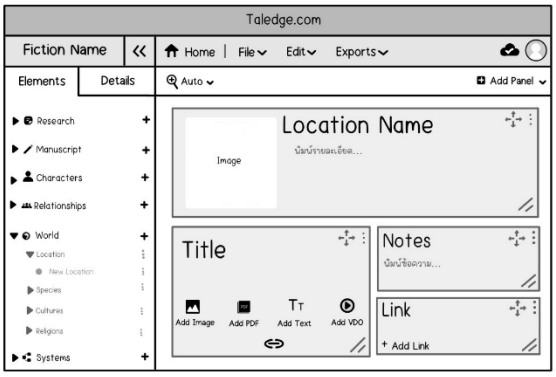
หน้า Relationship A group relationship web



หน้า Relationship New relationship web ความสัมพันธ์ระหว่างสอง group

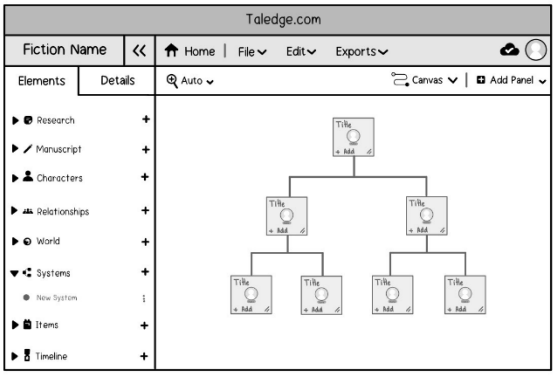


หน้า World building

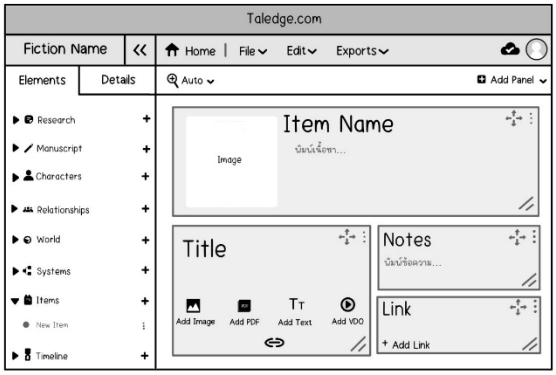


หน้า Systems

ตัวอย่าง class system

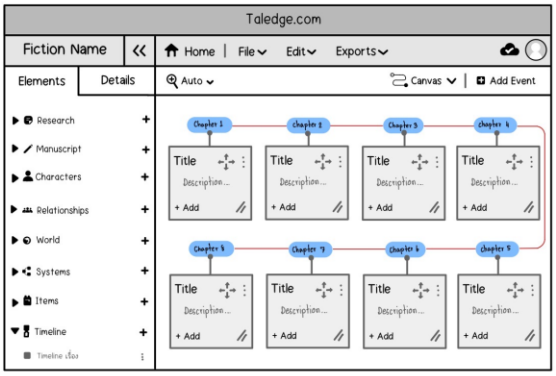


หน้า Item

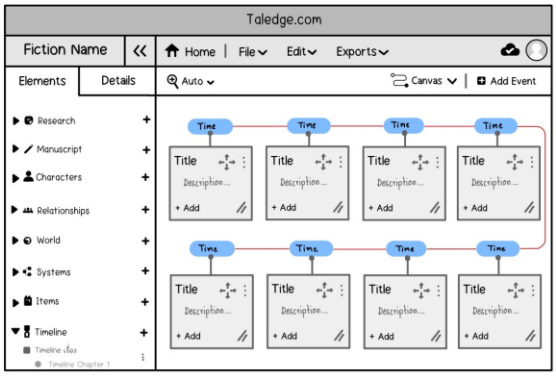


หน้า Timeline

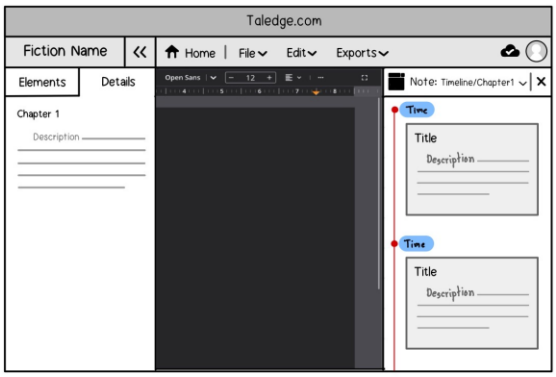
ตาง timeline เรื่อง



ตาง timeline chapter



เชื่อมไปหน้าเขียนฉาก



3.4 ประเด็นที่น่าสนใจและท้าทาย

3.4.1 ประเด็นที่น่าสนใจ

1. การเชื่อมโยงองค์ประกอบต่างๆ ของนิยายให้เห็นภาพรวมชัดเจน เช่น การแสดงความสัมพันธ์ของตัวละครและไทม์ไลน์
2. การออกแบบฐานข้อมูลที่รองรับความซับซ้อนของตัวละครและเหตุการณ์
3. การแสดงผลโครงสร้างพล็อตในรูปแบบที่เข้าใจง่าย

3.4.2 สิ่งที่ท้าทาย

1. การพัฒนา UI/UX ที่ตอบสนองต่อความต้องการของผู้ใช้งานที่มีวิธีการเขียนที่หลากหลาย
2. การซิงค์ข้อมูลข้ามแพลตฟอร์มอย่างราบรื่น
3. การออกแบบฐานข้อมูลที่สามารถจัดการข้อมูลจำนวนมาก เช่น โครงสร้างพล็อตและข้อมูลตัวละครที่ซับซ้อน

3.5 ผลลัพธ์ที่คาดหวัง

3.5.1 ระบบสามารถทำงานได้ครอบคลุมทุกกรณีการใช้งาน (Use Case)

1. **การสร้างและจัดการพล็อตนิยาย** ระบบจะช่วยให้นักเขียนสามารถสร้างและปรับแต่งพล็อตได้อย่างยืดหยุ่น รวมถึงจัดการตัวละคร สถานที่ และไทม์ไลน์ได้อย่างครบถ้วน
2. **การเชื่อมโยงองค์ประกอบ** ฟังก์ชันการเชื่อมโยงตัวละครและเหตุการณ์เพื่อให้เกิดภาพรวมที่ชัดเจน
3. **การจัดการเนื้อหา** ระบบช่วยในการเพิ่ม แก้ไข และบันทึกเนื้อหาต้นฉบับ (manuscript) ได้สะดวก

3.5.2 ระบบสามารถทำงานได้ตาม Non-functional Requirements

1. ระบบมีอินเทอร์เฟซที่ใช้งานง่าย ช่วยลดเวลาในการเรียนรู้และเพิ่มความสะดวกในการใช้งาน
2. ระบบสามารถรองรับการใช้งานบนอุปกรณ์หลากหลายประเภท เช่น คอมพิวเตอร์ แท็บเล็ต และสมาร์ทโฟน
3. ข้อมูลถูกบันทึกและซิงค์อย่างปลอดภัย รองรับการใช้งานข้ามอุปกรณ์โดยไม่มีปัญหาเกี่ยวกับการสูญหายของข้อมูล

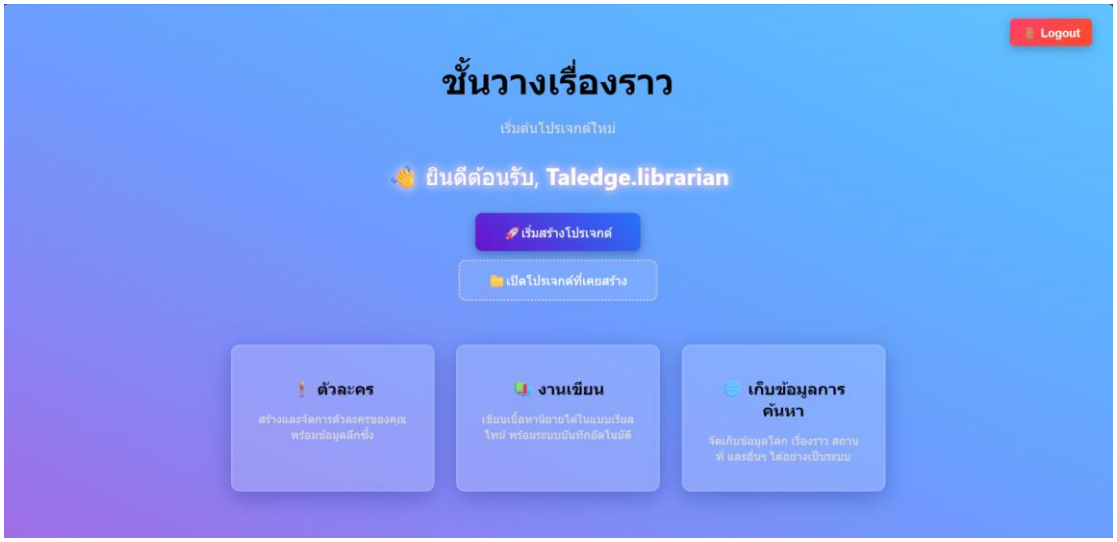
บทที่ 4
ผลการดำเนินงาน

โครงงานนี้ได้ทำการพัฒนาเว็บแอปพลิเคชัน TALEDGE ซึ่งประกอบด้วยฟีเจอร์หลักในการช่วยนักเขียนในการวางพล็อตนิยาย ได้แก่ การจัดการตัวละคร (Characters), เหตุการณ์และไทม์ไลน์ (Timeline), ไอเทมในเรื่อง (Items), ต้นฉบับ (Manuscript), ระบบเก็บข้อมูลไอเดีย (Research), ระบบสร้างโลกนิยาย (World) และระบบผังความสัมพันธ์ชนชั้น (System) โดยใช้เทคโนโลยี React.js ในส่วน Front-end และ Node.js + MongoDB ในส่วน Back-end

4.1 หน้าหลัก (Home)

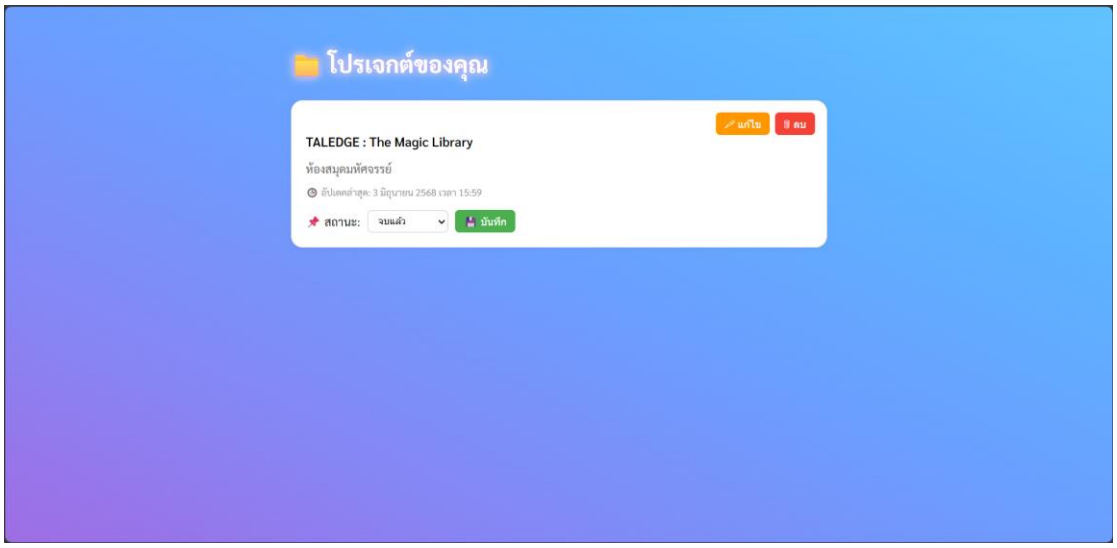
เมื่อผู้ใช้เข้าสู่ระบบ จะพบกับหน้า Home ซึ่งประกอบด้วยปุ่ม:

- **“สร้างโปรเจกต์ใหม่”** เพื่อเริ่มแต่งนิยายเรื่องใหม่
- **“เปิดโปรเจกต์ที่เคยสร้าง”** สำหรับเข้าถึงเรื่องเก่าที่เคยสร้างไว้



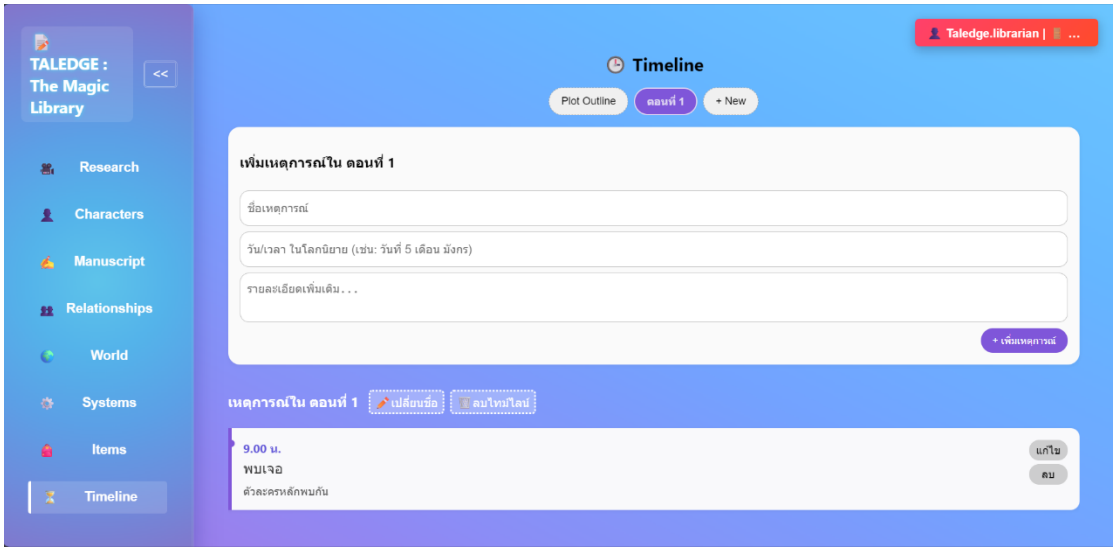
4.2 โปรเจกต์ที่เคยสร้าง

แสดงรายการโปรเจกต์ทั้งหมดที่เคยสร้าง ผู้ใช้สามารถ ลบ/เปิด/แก้ไข โปรเจกต์ได้



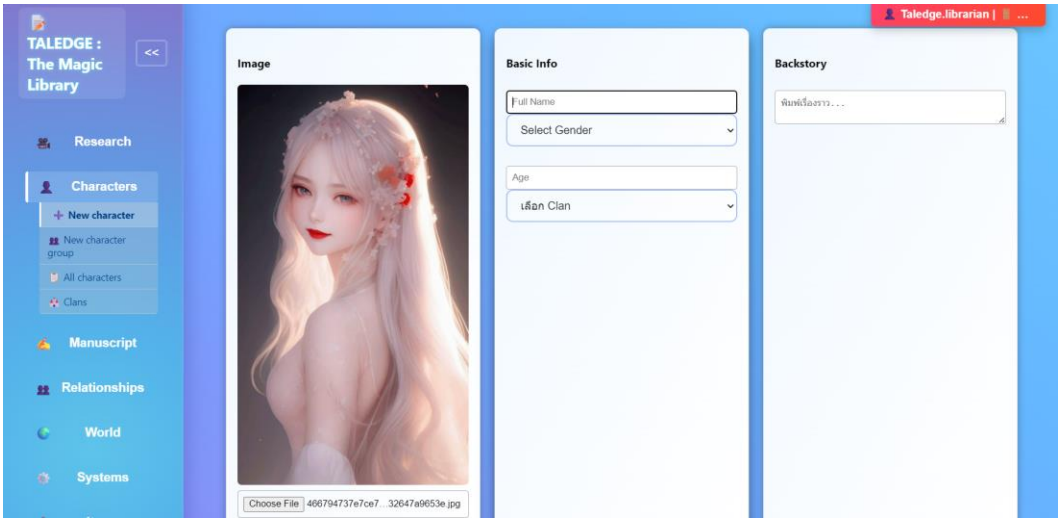
4.3 หน้าสร้างพล็อตเรื่องและจัดการเหตุการณ์ (Timeline)

ในหน้านี้ผู้ใช้สามารถเพิ่ม Timeline ของเรื่องราว แก้ไข หรือจัดเรียงเหตุการณ์ตามลำดับเวลาได้อย่างเป็นระบบ



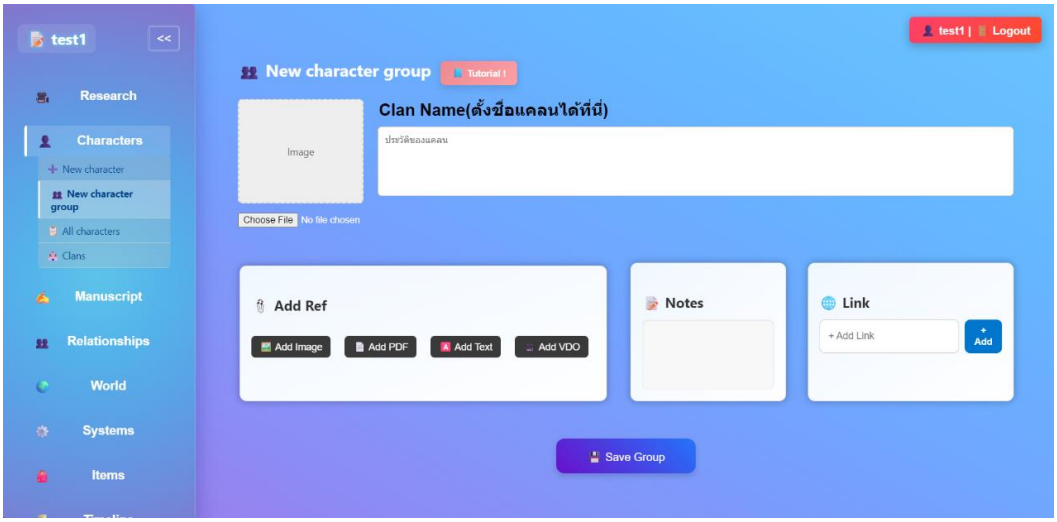
4.4 การจัดการตัวละคร (Characters Panel)

ผู้ใช้สามารถเพิ่มตัวละครใหม่ พร้อมกรอกข้อมูลสำคัญ เช่น ชื่อ อายุ ลักษณะภายนอก บุคลิก และความสัมพันธ์กับตัวละครอื่น ๆ ระบบมีการออกแบบให้รองรับการแสดงความเชื่อมโยงระหว่างตัวละครผ่าน Diagram แต่ในปัจจุบัน ระบบยังไม่สามารถแสดงความเชื่อมโยงระหว่างตัวละครแบบอัตโนมัติได้ ผู้ใช้ต้องเพิ่มความสัมพันธ์ด้วยตนเอง



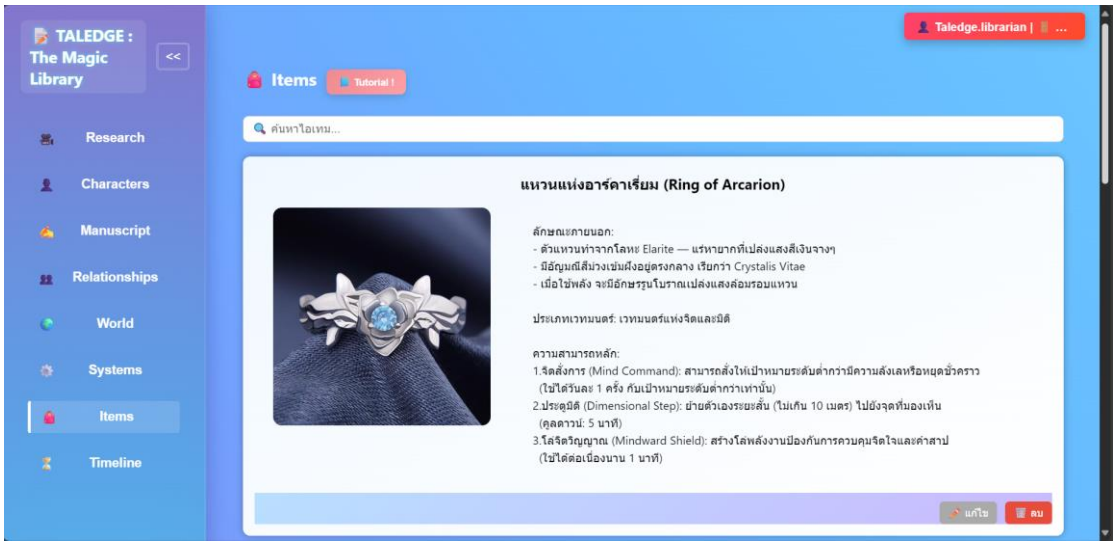
4.5 การจัดกลุ่มตัวละคร (Character Groups)

ช่วยให้สามารถรวมตัวละครไว้เป็นกลุ่มหรือองค์กร เช่น เผ่าพันธุ์ กลุ่มนักรบ สมาชิกในโรงเรียน ฯลฯ



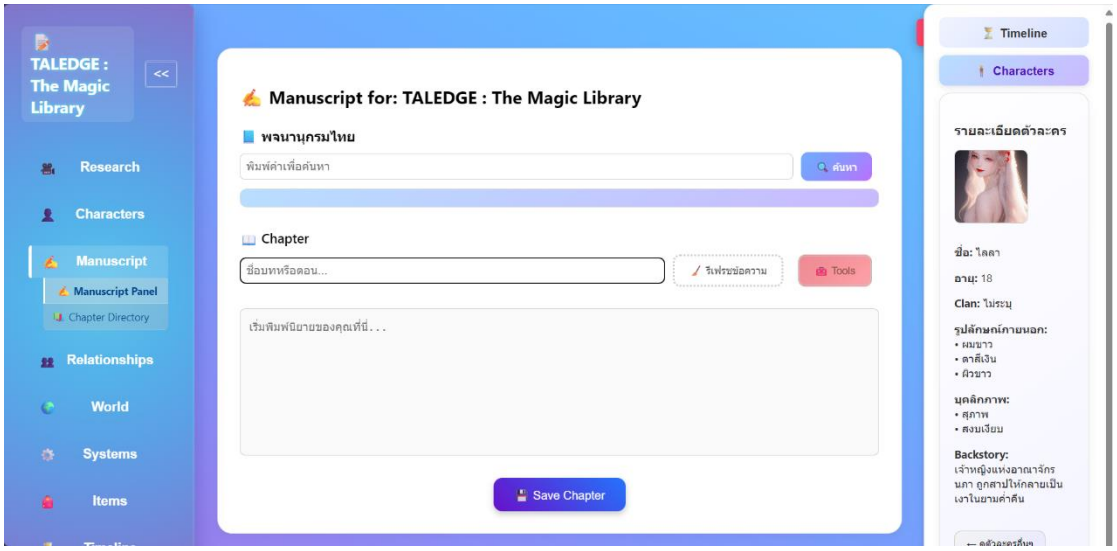
4.6 การจัดการไอเทมในเรื่อง (Items Panel)

ระบบรองรับการเพิ่มไอเทม เช่น อาวุธ สิ่งของวิเศษ เอกสาร ฯลฯ ผู้ใช้สามารถแนบรูปภาพ คำอธิบาย และลิงก์เอกสารหรือลิงก์วิดีโอเพิ่มเติมได้ในแต่ละไอเทม



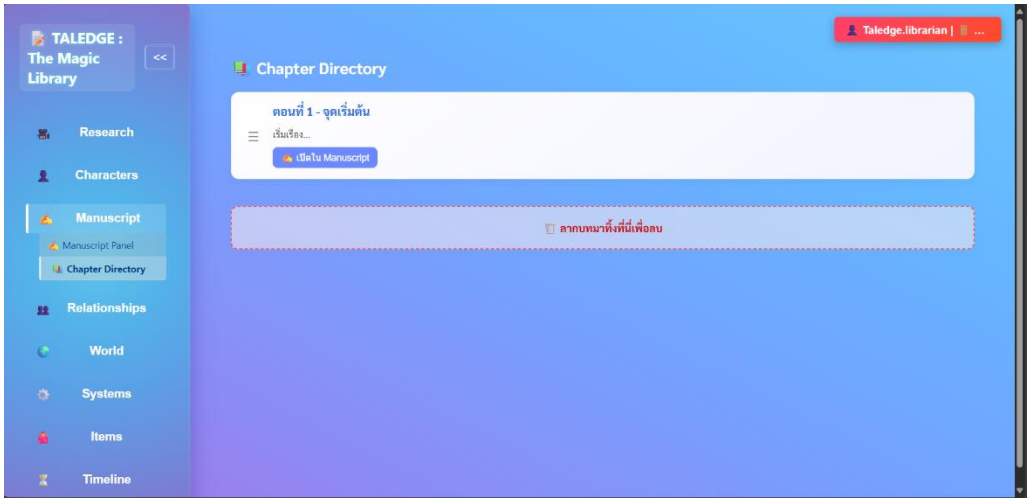
4.7 การเขียนต้นฉบับ (Manuscript Panel)

ผู้ใช้สามารถเริ่มต้นเขียนเนื้อหานิยายได้โดยตรงในระบบ โดยมี sidebar สำหรับแสดงข้อมูล Timeline และ Character ที่สร้างไว้มาช่วยอ้างอิงในการเขียน ระบบยังไม่มีฟีเจอร์ autosave ผู้ใช้ต้องกดบันทึกเองเมื่อเขียนเสร็จ



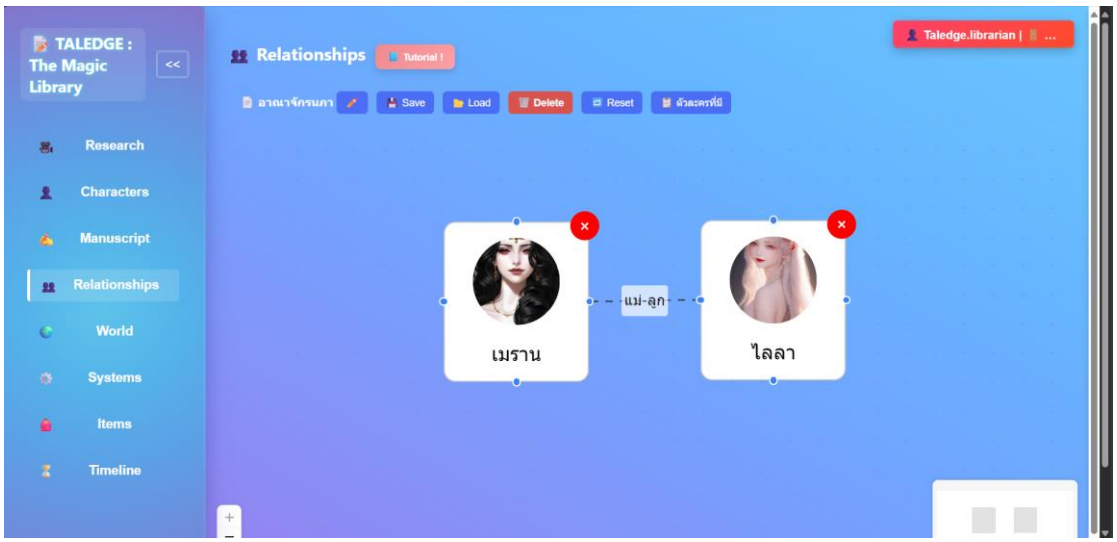
4.8 การจัดการบท (Chapter Directory)

- แยกเนื้อหาออกเป็นบท (Chapters)
- จัดการบทแบบ drag-and-drop ได้



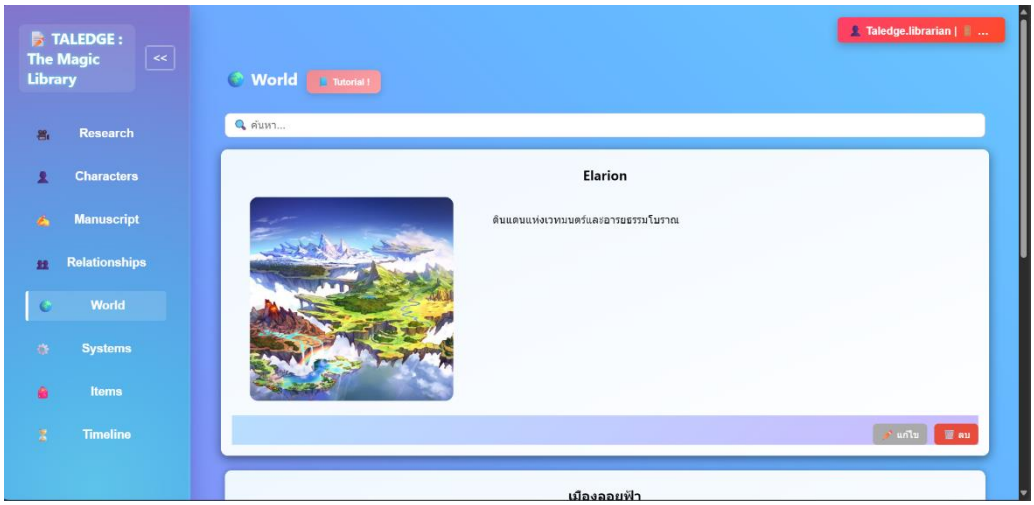
4.9 การเชื่อมโยงความสัมพันธ์รหว่างตัวละคร

TALEDGE ได้พัฒนาให้สามารถเชื่อมโยงความสัมพันธ์รหว่างตัวละคร โดยมีการใช้เทคนิค drag-and-drop ตัวละครที่ได้จากตัวละครที่สร้างไว้มาใช้เพื่อความสะดวก



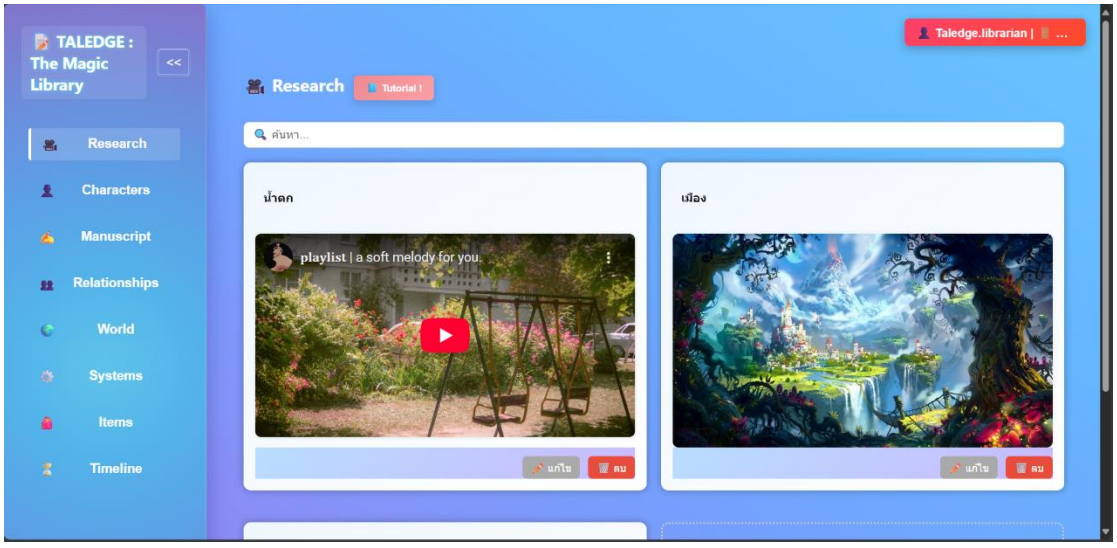
4.10 การจัดการโลกในเรื่อง (World Panel)

สร้างโลกนิยายของคุณเอง สร้างสถานที่ต่างๆ โดยผู้ใช้สามารถแนบรูปภาพ คำอธิบาย และลิงก์เอกสารหรือลิงก์วิดีโอเพิ่มเติมได้ในแต่ละการ์ดสถานที่



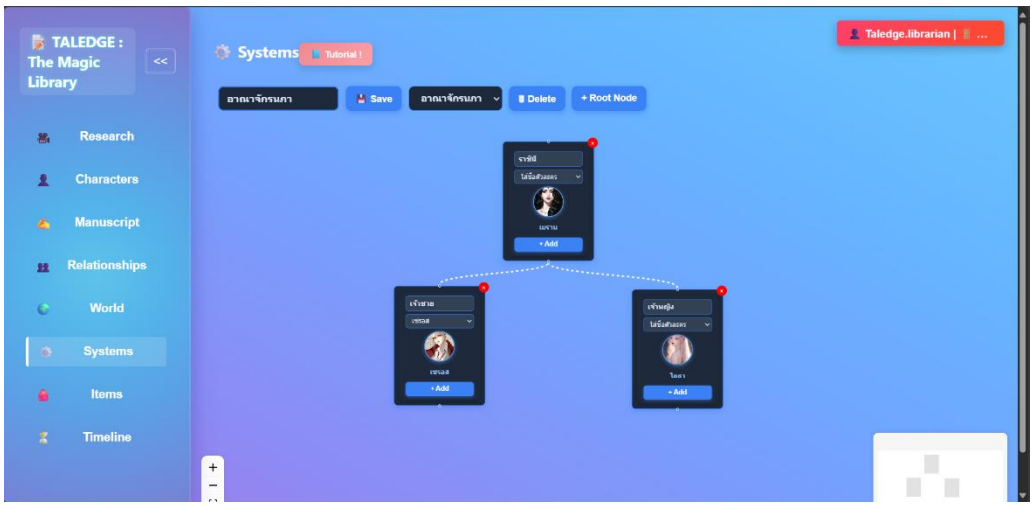
4.11 การจัดการไอเดียหรือการอ้างอิง (Research Panel)

ระบบ Research ออกแบบมาเพื่อให้ผู้ใช้สามารถเก็บข้อมูลอ้างอิงหรือไอเดียต่างๆ ได้อย่างเป็นระเบียบ เช่น การวางลิงก์, รูปภาพ, ไฟล์ PDF หรือบันทึกข้อความสั้นๆ สำหรับใช้อ้างอิงในการแต่งนิยาย



4.12 ระบบ System (ผังระบบความสัมพันธ์หรือชนชั้น)

ระบบนี้ช่วยสร้างแผนผังของโครงสร้างสังคม เช่น ระบบชนชั้น องค์กร กลุ่มทางการเมือง หรือเผ่าพันธุ์ โดยใช้โครงสร้างแบบแผนภาพ เพื่อแสดงความสัมพันธ์แบบมีลำดับขั้น



บทที่ 5
สรุป

โครงงาน "TALEDGE: ชั้นวางเรื่องราว" ได้พัฒนาเว็บแอปพลิเคชันที่ช่วยให้นักเขียนสามารถวางพล็อตนิยาย จัดระเบียบตัวละคร เหตุการณ์ และเนื้อหาได้อย่างเป็นระบบ โดยมุ่งเน้นความใช้งานง่าย รองรับกระบวนการคิดสร้างสรรค์ และจัดการองค์ประกอบต่าง ๆ ของนิยายได้ในที่เดียว ไม่ว่าจะเป็นการออกแบบไทม์ไลน์ การสร้างโลก การเก็บข้อมูลอ้างอิง และการเขียนต้นฉบับ ทั้งหมดนี้ถูกรวมเข้าไว้ในแพลตฟอร์มเดียวเพื่อสร้างประสบการณ์ที่ราบรื่นและเป็นมิตรต่อผู้ใช้

นอกจากนี้ ตัวระบบยังถูกออกแบบมาให้สามารถขยายฟีเจอร์เพิ่มเติมได้ในอนาคต เช่น ระบบทำงานร่วมกัน (Co-writing) หรือการใช้ AI ช่วยเขียน ทำให้ TALEDGE มีศักยภาพในการเติบโตเป็นเครื่องมือที่ครอบคลุมสำหรับนักเขียนยุคใหม่ทั้งในเชิงสร้างสรรค์และเทคโนโลยี

5.1 สรุปผลการดำเนินงาน

โครงงานนี้แสดงให้เห็นถึงความสามารถในการออกแบบและพัฒนาเว็บแอปพลิเคชันที่สามารถช่วยเหลือนักเขียนในกระบวนการวางพล็อตนิยายได้อย่างมีประสิทธิภาพ จากการออกแบบระบบที่ครอบคลุมตั้งแต่การวางพล็อต การจัดการตัวละคร การเขียนต้นฉบับ ไปจนถึงระบบรองรับองค์ประกอบอื่น ๆ อย่าง Research, World และ System พบว่าผู้ใช้สามารถทำงานได้สะดวกและต่อเนื่องมากยิ่งขึ้น

อย่างไรก็ตาม ในขณะนี้ระบบยังได้ผ่านการทดสอบเฉพาะบนอุปกรณ์คอมพิวเตอร์เท่านั้น และยังไม่ได้มีการนำไปทดสอบกับกลุ่มเป้าหมายจริง จึงยังไม่สามารถสรุปผลการใช้งานในบริบทของผู้ใช้ปลายทางได้อย่างครบถ้วน ซึ่งเป็นข้อจำกัดที่ควรพิจารณาเพิ่มเติมในการพัฒนาและประเมินผลในอนาคต

- ระบบสามารถตอบสนองตาม Use Case ที่กำหนดในบทที่ 3 ได้ครบถ้วน ทั้งในแง่ของฟีเจอร์หลักและประสบการณ์ผู้ใช้
- ผู้ใช้งานสามารถสร้างโปรเจกต์ จัดการข้อมูล และเขียนเนื้อหาได้อย่างมีประสิทธิภาพ ทั้งในด้านการใช้งานและการประมวลผล

- ระบบมีความยืดหยุ่น สามารถขยายความสามารถต่อในอนาคต ทั้งด้านเทคนิคและประสบการณ์ผู้ใช้
- ผู้ใช้งานสามารถสร้างโปรเจกต์ จัดการข้อมูล และเขียนเนื้อหาได้อย่างมีประสิทธิภาพ

5.2 ข้อเสนอแนะและแนวทางการพัฒนาต่อไป

1. เพิ่มระบบ "Co-writing" เพื่อให้ผู้ใช้งานสามารถทำงานร่วมกันในโปรเจกต์เดียวได้
2. พัฒนา AI Assistant สำหรับช่วยเสนอชื่อพล็อต หรือช่วยเขียนต่อ
3. ปรับปรุง UI ให้สามารถปรับธีม (Dark/Light Mode)
4. รองรับการ export ไฟล์เป็นรูปแบบ PDF หรือ ePub สำหรับการนำไปใช้งานต่อ
5. เพิ่มระบบ autosave เพื่อให้เนื้อหาถูกบันทึกอัตโนมัติ
6. เพิ่มพจนานุกรมภาษาไทย เพื่อช่วยนักเขียนในการค้นหาคำศัพท์และตรวจการสะกด
7. เพิ่มให้สามารถ Edit Clan เพื่อให้สามารถแก้ไขรายละเอียดของกลุ่มที่เคยสร้าง

รายการอ้างอิง

9Expert Training. (2022). *Python คืออะไร?*

สืบค้นจาก: https://www.9experttraining.com/articles/python-%E0%B8%84%E0%B8%B7%E0%B8%AD%E0%B8%AD%E0%B8%B0%E0%B9%84%E0%B8%A3

Amazon Web Services (AWS). (2022). *API คืออะไร?*

สืบค้นจาก: https://aws.amazon.com/th/what-is/api/

AppMaster. (2022). *การแบ่งปันทรัพยากรข้ามแหล่งกำเนิด (CORS).*

สืบค้นจาก: https://appmaster.io/th/glossary/kaaraebngpanthraphyaakrkhaamaehlngkamenid-cors

BorntoDev. (2021). *มาทำความรู้จักกับ PyThaiNLP.*

สืบค้นจาก: https://www.borntodev.com/2021/09/06/มาทำความรู้จักกับ-pythainlp/

Bualabs. (2021). *Flask คืออะไร? และการสร้าง Hello World App อย่างง่าย.*

สืบค้นจาก: https://www.bualabs.com/archives/3934/what-is-flask-tutorial-how-to-build-hello-world-app-python-install-flask-framework-deploy-on-heroku-by-example-heroku-ep-2/

Chanin Chongmeesuk. (2020). *ใช้ MongoDB ฟรีๆ ด้วย MongoDB Atlas.*

สืบค้นจาก: https://medium.com/@chaninchongmeesuk/ใช้-mongodb-ฟรีๆ-ด้วย-mongodb-atlas-c16ca21d8f34

Foxbith. (2021). *Web Programming คืออะไร? ภาษาที่ต้องรู้มีอะไรบ้าง.*

สืบค้นจาก: https://www.foxbith.com/blog/what-is-web-programming

Foxbith. (2021). *[คู่มือ] การพัฒนาแอปพลิเคชันทุกขั้นตอนตั้งแต่พื้นฐาน.*

สืบค้นจาก: https://shorturl.asia/y29ir

MongoDB. (2021). *เริ่มต้นกับ MongoDB Atlas [YouTube Video].*

สืบค้นจาก: https://www.youtube.com/watch?v=KtTsXi_8he0

Relevant Audience. (2021). *CMS คืออะไร และทำไมถึงสำคัญต่อเว็บไซต์ของคุณ?*

สืบค้นจาก: https://www.relevantaudience.com/th/what-is-a-cms-and-why-is-it-important-to-make-a-website/

Skooldio. (2022). *UX/UI Designer คืออะไร? ทำไมถึงเป็นที่ต้องการสูง?*

สืบค้นจาก: https://blog.skooldio.com/ux-ui-designer-ultimate-guide/

Super AI Engineer (Medium). (2022). *การใช้งาน Library NLTK เบื้องต้น.*

สืบค้นจาก: https://medium.com/super-ai-engineer/การใช้งาน-library-nltk-เบื้องต้น-df2a422c4b06

The Enterprise (Medium). (2021). *Translation API วุ้นแปลภาษาสำหรับองค์กรจาก Google*. สืบค้นจาก: https://medium.com/the-enterprise/translation-api-วุ้นแปลภาษาสำหรับองค์กรจาก-google-8cd658dcf2c9

The Ohio State University. (2021). *What is Narrative Theory?*

สืบค้นจาก: https://projectnarrative.osu.edu/about/what-is-narrative-theory

มหาวิทยาลัยเทคโนโลยีราชมงคลพระนคร. (2020). *บทที่ 2 หลักการและทฤษฎีที่เกี่ยวข้อง.*

สืบค้นจาก: https://sci.rmutp.ac.th/web2558/wp-content/uploads/2020/05/06_ch2-2.pdf

ภาคผนวก

โครงสร้างของ Source Code ที่น่าสนใจและท้าทาย

ภาคผนวก ก. ส่วนของ Relationships หรือ แผนผังความสัมพันธ์

แสดงผลแบบ Interactive Graph ด้วย React Flow

ตัวอย่าง Code: การสร้าง Node ของแต่ละ Node ให้ลากเส้นหากันได้ทุกจุด

```
{/* TOP */}
<Handle type="target" position="top" id="top" style={{ left: "50%", transform: "translateX(-50%)", background: "#555" }} />
<Handle type="source" position="top" id="top" style={{ left: "50%", transform: "translateX(-50%)", background: "#555" }} />

{/* BOTTOM */}
<Handle type="target" position="bottom" id="bottom" style={{ left: "50%", transform: "translateX(-50%)", background: "#555" }} />
<Handle type="source" position="bottom" id="bottom" style={{ left: "50%", transform: "translateX(-50%)", background: "#555" }} />

{/* LEFT */}
<Handle type="target" position="left" id="left" style={{ top: "50%", transform: "translateY(-50%)", background: "#555" }} />
<Handle type="source" position="left" id="left" style={{ top: "50%", transform: "translateY(-50%)", background: "#555" }} />

{/* RIGHT */}
<Handle type="target" position="right" id="right" style={{ top: "50%", transform: "translateY(-50%)", background: "#555" }} />
<Handle type="source" position="right" id="right" style={{ top: "50%", transform: "translateY(-50%)", background: "#555" }} />
```

ตัวอย่าง Code : การเรียกใช้ Library ในการทำกราฟ และ ทำให้ Node แต่ละ Node ทำงานได้

```
<div
  ref={reactFlowWrapper}
  style={{ width: "300%", height: "200%" }}
  onDrop={onDrop}
  onDragOver={onDragOver}
>
  <ReactFlow
    nodes={nodes}
    edges={edges}
    onNodesChange={onNodesChange}
    onEdgesChange={onEdgesChange}
    onConnect={onConnect}
    onEdgeDoubleClick={onEdgeDoubleClick}
    fitView
    nodeTypes={nodeTypes}
    connectionMode="loose"
  >
    <MiniMap className="fixed-minimap" pannable zoomable />
    <Controls className="fixed-controls" />
    <Background />
  </ReactFlow>
</div>
```

ภาคผนวก ข. ส่วนของ System หรือ ระบบชนชั้น

ตัวอย่าง Code: การจัดการ Node แบบ Dynamic และ Nested Component Update

```
const handleTitleChange = (nodeId, newTitle) => {
  setNodes((prev) =>
    prev.map((n) =>
      n.id === nodeId
        ? {
            ...n,
            data: {
              ...n.data,
              label: (
                <EditableTitleNode
                  nodeId={nodeId}
                  title={newTitle}
                  character={n.data.label?.props?.character || null}
                  onAddCharacter={handleAttachCharacter}
                  onCreateChild={handleAddNode}
                  onTitleChange={handleTitleChange}
                  characters={Array.isArray(charList) ? charList : []}
                />
              ),
            },
```

- ReactFlow ใช้ data.label เป็น field รับ JSX ได้ แต่ useState() ไม่สามารถ track React component internals ได้ (ถ้าใช้ function/component) จึงต้องระวัง state loss และ rerender manual ทำให้ทุกครั้งที่ต้องการเปลี่ยน title/character ต้อง recreate ทั้ง label JSX ใหม่
- เสี่ยง bug จาก props ไม่ตรง
- ต้องเขียน handleTitleChange, handleAttachCharacter, handleAddNode ให้เก่งพอที่จะ map ข้อมูลกลับเข้าระบบโดยไม่ทำให้ node เสีย structure
- ทำให้การ serialize/deserialize (ตอน save/load) ต้องเขียนโค้ดแปลงเอง (เช่น sanitizeNodes) ให้แยก component ออก

ต่อมาจะเป็นในส่วนของระบบการเชื่อมโยงระหว่าง parent กับ child ผ่าน character selector

- ต้องทำ logic กรองว่า character ไหนยังว่าง
- ต้องสร้าง edge เชื่อมทันทีเมื่อกด confirm ซึ่งการโหลดและ Restore node จาก MongoDB แล้ว rebuild JSX (ใน handleLoad)
- ต้อง rebuild label จาก JSON ที่เคย serialize ไปแล้ว
- หาก field หรือ character หาย จะพังได้ง่ายๆ

ภาคผนวก ค. การใช้ API Dictionary app.py

ตัวอย่าง Code : Front-end การ ตรวจสอบคำค้นหา: ถ้า dictionaryTerm ว่างจะไม่ทำงาน เรียก API: ส่ง HTTP GET ไปยัง backend URL:

```
const handleLookup = async () => {
  if (!dictionaryTerm.trim()) return;

  try {
    const res = await fetch(`http://127.0.0.1:5000/lookup?word=${dictionaryTerm}`);
    const data = await res.json();
    const meaning = `🗣️ คำแปล: ${data.translation}
📘 ความหมาย: ${data.definition}`;
    setDefinition(meaning);
  } catch {
    setDefinition("❌ ไม่สามารถเชื่อมต่อ API ได้");
  }
};
```

ตัวอย่าง Code : Back – End ข้อเสียคือคำแปลที่ได้คือภาษาอังกฤษ

```
@app.route("/lookup", methods=["GET"])
def lookup():
    word = request.args.get("word", "").strip()
    if not word:
        return jsonify({"error": "❌ โปรดใส่คำค้นหา"}), 400

    try:
        lang = "en" if word.isascii() else "th"
        synset = synsets(word, lang="tha" if lang == "en" else "tha")
        definitions = [s.definition() for s in synset] if synset else []

        translated = translate(word, "th" if lang == "en" else "en")
        meaning = definitions[0] if definitions else "ไม่พบคำนิยามใน WordNet"

        return jsonify({
            "translation": translated,
            "definition": meaning
        })
    except Exception as e:
        return jsonify({"definition": f"❌ เกิดข้อผิดพลาด: {str(e)}"}), 500

if __name__ == "__main__":
    app.run(port=5000)
```